# PATENT ABSTRACTS OF JAPAN

(11)Publication number : 2000-182012
(43)Date of publication of application : 30.06.2000

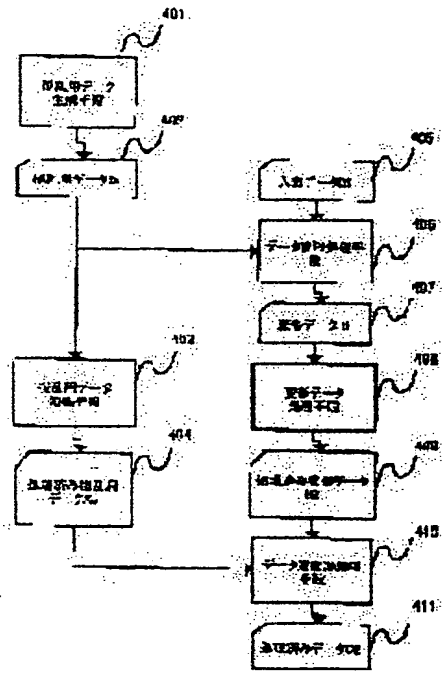
(51)Int.Cl. G06K 19/073
G09C 1/00
H04L 9/10

(21)Application number : 10-354156
(22)Date of filing : 14.12.1998

(71)Applicant : HITACHI LTD
(72)Inventor : OKI MASARU
FUKUZAWA YASUKO
OKUHARA SUSUMU
KAMINAGA MASAHIRO

## (54) INFORMATION PROCESSOR AND END TAMPER PROCESSOR

(57)Abstract:
PROBLEM TO BE SOLVED: To make it possible to interence processing and a ciphering key from the waveform of current consumption by deforming processing data with disturbing data, processing data with deformed data and inversely deforming it through the use of disturbing data after processing to obtain a correct processing result in an IC card chip so as to reduce relation between data processing and current consumption in the IC card chip.
SOLUTION: A disturbing data generating means 401 generates disturbing data Xi. Next, a data deforming processing means 406 deforms an inputted data D1 (405) with disturbing data Xi to generate deforming data H1 (407). Then, through the use of data H1, a deformed data processing means 408 executes data processing to generate processed deforming data H2 (409). On the other hand, the data Xi is given data processing equal to inputted data by a disturbing data processing means 403 to prepare processed disturbing data Xo (404). True processed data D2 (411) is obtained by a data inversely deform processing means 410 through the use of this data Xo and data H2.




LEGAL STATUS

[Date of request for examination]

[Date of sending the examiner's decision of rejection]

[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

[Date of final disposal for application]

[Patent number]

[Date of registration]

[Number of appeal against examiner's decision of rejection]

[Date of requesting appeal against examiner's decision of rejection]

[Date of extinction of right]

(19)日本国特許庁（ＪＰ）　(12)公開特許公報（Ａ）　(11)特許出願公開番号
特開2000－182012
（P2000－182012A）
(43)公開日　平成12年６月30日(2000.6.30)

| (51)Int.Cl.' | 識別記号 | ＦＩ | | テーマコード'（参考） |
|---|---|---|---|---|
| Ｇ０６Ｋ　19/073 | | Ｇ０６Ｋ　19/00 | Ｐ | ５Ｂ０３５ |
| Ｇ０９Ｃ　1/00 | ６６０ | Ｇ０９Ｃ　1/00 | ６６０Ａ | ５Ｊ１０４ |
| Ｈ０４Ｌ　9/10 | | Ｈ０４Ｌ　9/00 | ６２１Ｚ | ９Ａ００１ |

審査請求　未請求　請求項の数23　ＯＬ　（全 26 頁）

(21)出願番号　特願平10－354156

(22)出願日　平成10年12月14日(1998.12.14)

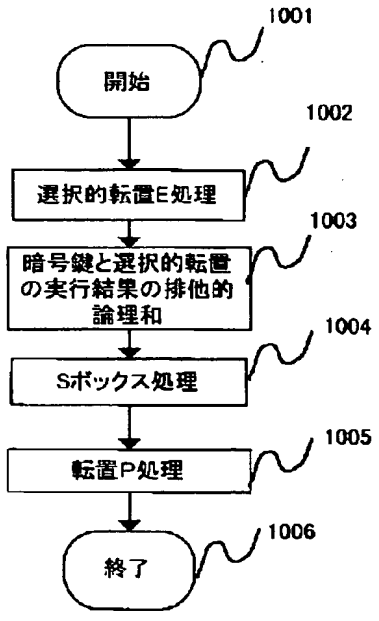
(71)出願人　000005108
株式会社日立製作所
東京都千代田区神田駿河台四丁目６番地
(72)発明者　大木　優
東京都国分寺市東恋ケ窪一丁目280番地
株式会社日立製作所中央研究所内
(72)発明者　福澤　寧子
神奈川県川崎市麻生区王禅寺1099番地　株式会社日立製作所システム研究所内
(74)代理人　100068504
弁理士　小川　勝男

最終頁に続く

(54)【発明の名称】　情報処理装置、端タンパ処理装置

(57)【要約】
【課題】　ICカード用チップのプログラムでの処理を行う前に変形を加え,処理が終わった後に逆変形を行い,正しい処理結果を得る。ICカード用チップでの処理内容とICカード用チップの消費電流との関連性を減らす。
【解決手段】　ICカード用チップの処理を行う前にを行う前に変形を加えることにより,処理データとICカード用チップの消費電流との関連性を減らす。処理が終わった後に逆変形を行い,正しい処理結果を得る。ICカード用チップでの処理のデータとICカード用チップの消費電流との関連性を減らすことを可能とする。

図10



【特許請求の範囲】

【請求項１】プログラムを格納するプログラム格納部、データを保存するデータ格納部を持つ記憶装置とプログラムに従い、所定の処理を実行し、データ処理を行う中央演算装置を持ち、プログラムは、中央演算装置に実行の指示を与える処理命令から構成される一つ以上のデータ処理手段からなり、一つのデータ処理手段が、その入力データを処理し、処理済みデータを出力する入力データ処理手段を含む、情報処理装置において、撹乱用データXiを使って入力データD1を変形し、変形データH1を作成するデータ変形処理手段、変形データH1を入力データ処理手段と同じ処理を行い、処理済み変形データH2を作成する変形データ処理手段、撹乱用データXiを入力データ処理手段と同じ処理を行い、処理済み撹乱用データXoを作成する撹乱用データ処理手段、処理済み撹乱用データXoを使って処理済み変形データH2を処理し、入力データD1を入力データ処理手段で処理した結果である処理済みデータD2を得るデータ逆変形処理手段、を有することを特徴とする情報処理装置。

【請求項２】プログラムを格納するプログラム格納部、データを保存するデータ格納部を持つ記憶装置とプログラムに従い、所定の処理を実行し、データ処理を行う中央演算装置を持ち、プログラムは、中央演算装置に実行の指示を与える処理命令から構成される一つ以上のデータ処理手段からなり、一つのデータ処理手段が、その第１入力データを処理し、第１処理済みデータを出力する第１入力データ処理手段と、別のデータ処理手段が、その第２入力データを処理し、第２処理済みデータを出力する第２入力データ処理手段を含む、情報処理装置において、第１撹乱用データX1iを使って第１入力データD1を変形し、変形データH1を作成する第１データ変形処理手段、変形データH1を第１入力データ処理手段と同じ処理を行い、処理済み変形データH2を作成する第１変形データ処理手段、第１撹乱用データX1iを第１入力データ処理手段と同じ処理を行い、第１処理済み撹乱用データX1oを作成する第１撹乱用データ処理手段、第２撹乱用データX2iを使って処理済み変形データH2を変形し、処理済み変形データH3を作成する第２データ変形処理手段、処理済み変形データH3を第２入力データ処理手段と同じ処理を行い、処理済み変形データH4を作成する第２変形データ処理手段、第２撹乱用データX2iを第２入力データ処理手段と同じ処理を行い、第２処理済み撹乱用データX2oを作成する第２撹乱用データ処理手段、第２処理済み撹乱用データX2oを使って処理済み変形データH4を処理し、処理済み変形データH5を得る第２データ逆変形処理手段、第１処理済み撹乱用データX1oを使って処理済み変形データH5を処理し、処理済みデータD2を得る第１データ逆変形処理手段、を有することを特徴とする情報処理装置。

【請求項３】プログラムを格納するプログラム格納部、データを保存するデータ格納部を持つ記憶装置とプログラムに従い、所定の処理を実行し、データ処理を行う中央演算装置を持ち、プログラムは、中央演算装置に実行の指示を与える処理命令から構成される一つ以上のデータ処理手段からなり、一つのデータ処理手段が、その第１入力データを処理し、第１処理済みデータを出力する第１入力データ処理手段と、別のデータ処理手段が、その第２入力データを処理し、第２処理済みデータを出力する第２入力データ処理手段を含む、情報処理装置において、第１撹乱用データX1iを使って第１入力データD1を変形し、変形データH1を作成する第１データ変形処理手段、変形データH1を第１入力データ処理手段と同じ処理を行い、処理済み変形データH2を作成する第１変形データ処理手段、第１撹乱用データX1iを第１入力データ処理手段と同じ処理を行い、第１処理済み撹乱用データX1oを作成する第１撹乱用データ処理手段、第２撹乱用データX2iを使って処理済み変形データH2を変形し、処理済み変形データH3を作成する第２データ変形処理手段、第１処理済み撹乱用データX1oを使って処理済み変形データH3を処理し、処理済み変形データH4を得る第１データ逆変形処理手段、処理済み変形データH4を第２入力データ処理手段と同じ処理を行い、処理済み変形データH5を作成する第２変形データ処理手段、第２撹乱用データX2iを第２入力データ処理手段と同じ処理を行い、第２処理済み撹乱用データX2oを作成する第２撹乱用データ処理手段、第２処理済み撹乱用データX2oを使って処理済み変形データH5を処理し、処理済みデータD2を得る第２データ逆変形処理手段、を有することを特徴とする情報処理装置。

【請求項４】プログラムを格納するプログラム格納部、データを保存するデータ格納部を持つ記憶装置とプログラムに従い、所定の処理を実行し、データ処理を行う中央演算装置を持ち、プログラムは、中央演算装置に実行の指示を与える処理命令から構成される一つ以上のデータ処理手段からなり、一つのデータ処理手段が、その第１入力データを処理し、第１処理済みデータを出力する第１入力データ処理手段と、別のデータ処理手段が、その第２入力データを処理し、第２処理済みデータを出力する第２入力データ処理手段を含む、情報処理装置において、第１撹乱用データX1iを使って第１入力データD1を変形し、変形データH1を作成する第１データ変形処理手段、変形データH1を第１入力データ処理手段と同じ処理を行い、処理済み変形データH2を作成する第１変形データ処理手段、第１撹乱用データX1iを第１入力データ処理手段と同じ処理を行い、第１処理済み撹乱用データX1oを作成する第１撹乱用データ処理手段、第２撹乱用データX2iを使って処理済み変形データH2を変形し、処理済み変形データH3を作成する第２データ変形処理手段、処理済み変形データH3を第２入力データ処理手段と同じ処理を行い、処理済み変形データH4を作成する第２

変形データ処理手段、第２撹乱用データX2iを第２入力データ処理手段と同じ処理を行い、第２処理済み撹乱用データX2oを作成する第２撹乱用データ処理手段、第１処理済み撹乱用データX1oと第２処理済み撹乱用データX2oの逆変形を統合し、統合処理済撹乱用データXoを作成するデータ逆変形統合処理手段、統合処理済撹乱用データXoを使って処理済み変形データH4を処理し、処理済みデータD2を得るデータ逆変形処理手段、を有することを特徴とする情報処理装置。

【請求項５】請求項１、２、３、４において、撹乱用データXiを入力データ処理手段と同じ処理を行い、処理済み撹乱用データXoを作成する撹乱用データ処理手段、処理済み撹乱用データXoを記録する処理済み撹乱用データ記録手段、記録した処理済み撹乱用データXoを使って処理済み変形データを処理し、新たに別の処理済み変形データを得るデータ逆変形処理手段、を有することを特徴とする情報処理装置。

【請求項６】請求項１、２、３、４、５において、撹乱用データとして乱数を生成する隠蔽データ生成処理手段、を有することを特徴とする情報処理装置。

【請求項７】請求項１、２、３、４、５、６において、データ変形処理手段およびデータ逆変形処理手段として、排他的論理和を使うことを特徴とする情報処理装置。

【請求項８】請求項１、２、３、４、５、６において、データ変形処理手段として、加算を使うことデータ逆変形処理手段として、減算を使うことを特徴とする情報処理装置。

【請求項９】請求項１、２、３、４、５、６において、データ変形処理手段として、減算を使うことデータ逆変形処理手段として、加算を使うことを特徴とする情報処理装置。

【請求項１０】請求項１、２、３、４、５、６において、データ変形処理手段として、掛け算を使うことデータ逆変形処理手段として、割り算を使うことを特徴とする情報処理装置。

【請求項１１】請求項１、２、３、４、５、６において、データ変形処理手段として、割り算を使うことデータ逆変形処理手段として、掛け算を使うことを特徴とする情報処理装置。

【請求項１２】請求項１、２、３、４、５、６において、データ変形処理手段として、剰余演算で剰余演算の法Ｎの整数倍を加算することデータ逆変形処理手段が不要であることを特徴とする情報処理装置。

【請求項１３】請求項１、２、３、４、５、６において、剰余演算で

$1 = X * Y \bmod N$

という関係のある数X、Yにおいてデータ変形処理手段として、剰余演算でＸを整数倍乗算することデータ逆変形処理手段として、データ変形処理手段でＸを乗算した回数分、Yを整数倍乗算することを特徴とする情報処理装置。

【請求項１４】請求項１、２、３、４、５、６において、データ変形処理手段として、 Nを剰余演算の法として、(N+1)/2を整数倍乗算すること、データ逆変形処理手段として、データ変形処理手段で(N+1)/2を乗算した回数分、２を整数倍乗算することを特徴とする情報処理装置。

【請求項１５】請求項１、２、３、４、５、６において、データ変形処理手段として、剰余演算で2を整数倍乗算することデータ逆変形処理手段として、データ変形処理手段で２を乗算した回数分、Nを剰余演算の法として、(N+1)/2を整数倍乗算することを特徴とする情報処理装置。

【請求項１６】請求項１、２、３、４、５、６において、データ変形処理手段として、配列のデータの配置を規則的な方法で変更すること、データ逆変形処理手段として、配列のデータの配置が変更された方法で配列のデータをアクセスすることを特徴とする情報処理装置。

【請求項１７】請求項１６において、データ変形処理手段での配列のデータの配置を規則的に変更方法として、配列のインデックス（引数）をある数で排他的論理和をとり、データの配置を入れ替えること、データ逆変形処理手段で配列のデータにアクセスする際、配列のインデックスをデータ変形処理手段で使用した数で排他的論理和をとり、変形後の配列のインデックスを作成し、アクセスすることを特徴とする情報処理装置。

【請求項１８】請求項１から６において、データ処理手段がビット単位のデータを入れ替える転置処理手段であること、データ変形処理手段及びデータ逆変形処理手段が排他的論理和であることを特徴とする情報処理装置。

【請求項１９】請求項１から６において、データ処理手段がバイト単位のデータを入れ替える換字処理手段であること、データ変形処理手段及びデータ逆変形処理手段が排他的論理和であることを特徴とする情報処理装置。

【請求項２０】請求項１から６において、データ処理手段がテーブルを使ってデータを入れ替える処理手段であること、データ変形処理手段及びデータ逆変形処理手段が排他的論理和であることを特徴とする情報処理装置。

【請求項２１】請求項３において、第１のデータ処理が、データ１とデータ２の排他的論理和を行いデータ３を生成する、排他的論理和処理手段であること、第２のデータ処理が、データ３を用いて、配列のインデックスを計算する配列アクセス処理手段であること、第１の変形処理手段として、第１の撹乱用データとデータの排他的論理和を使うこと、第１のデータ逆変形処理手段として、排他的論理和を使うこと、第２の変形処理手段として、第２の撹乱用データと配列のインデックスを排他的論理和をとること、第２のデータ逆変形処理手段として、排他的論理和を使うこと、を特徴とする情報処理装

置。

【請求項２２】請求項１、２、３、４、５、６において、データ変形処理手段として、剰余演算で剰余演算の法Ｎの整数倍に１を加えたものを乗算すること、データ逆変形処理手段が不要であることを特徴とする情報処理装置。

【請求項２３】データーを保存する記憶装置と、撹乱用データを生成する撹乱用データ生成手段と、上記データを上記撹乱用データを用いて変形して変形データを作成する変形データ作成手段と、上記変形データを処理する変形データ処理手段と、上記撹乱用データを処理する撹乱用データ処理手段と、処理後の上記変形データを処理後の上記撹乱用データを用いて逆変形する逆変形手段とを有することを特徴とする耐タンパ処理装置。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】本発明は、機密性の高いICカードなどの耐タンパ装置に関するものである。

【０００２】

【従来の技術】ICカードは、主に、勝手に書き換えられない情報の保持や秘密情報である暗号鍵を使ったデータの暗号化や暗号文の復号化を行うために使われる装置である。ICカードは、電源を持っていないため、リーダライタに差し込まれると、電源の供給を受け、動作可能となる。動作可能になると、リーダライタからコマンドを受けて、コマンドに従い、データの転送を行う。ICカードの一般的な解説は、オーム社出版電子情報通信学会編水沢順一著「ICカード」などにある。

【０００３】ICカードの構成は、図１に示すように、カード１０１の上に、ICカード用チップ１０２を搭載したものである。一般にICカードは、接点を持ち、接点を通して、リーダーライタから電源の供給やリーダライタとのデータの通信を行う。

【０００４】ICカード用チップの構成は、基本的にマイクロコンピュータと同じような構成である。その構成は、図２に示すように、中央演算装置２０１、記憶装置２０４、入出力ポート２０７、コ・プロセッサ２０２からなる。中央処理装置２０１は、論理演算や算術演算などを行う装置であり、記憶装置２０４は、プログラムやデータを格納する装置である。入出力ポートは、リーダライタと通信を行う装置である。コ・プロセッサは、剰余演算を行うための特別な演算装置であり、非対称暗号であるRSAの演算などに用いられる装置である。ICカード用プロセッサの中には、コ・プロセッサを持たないものも多くある。データバス２０３は、各装置を接続するバスである。

【０００５】記憶装置２０４は、ROM(Read Only Memory)やRAM(Random Access Memory)、EEPROM(Electrical Erasable Programmable Read Only Memory)などからなる。ROMは、変更できないメモリであり、主にプログラムを格納するメモリである。RAMは自由に書き換えができるメモリであるが、電源の供給が中断されると、記憶している内容が消えてなくなる。ICカードがリーダライタから抜かれると電源の供給が中断されるため、RAMの内容は、保持できなくなる。EEPROMは、電源の供給が中断されてもその内容を保持することができるメモリである。書き換える必要があり、ICカードがリーダライタから抜かれても、保持するデータを格納するために使われる。例えば、プリペイドカードでのプリペイドの度数などは、使用するたびに書き換えられ、かつリーダライタか抜かれてもデータを保持する必要があるため、EEPROMで保持される。

【０００６】ICカードは、プログラムや重要な情報がICカード用チップの中に密閉されているため、重要な情報を格納したり、カードの中で暗号処理を行うために、使われている。ICカードでの暗号処理の解読の難しさは、暗号アルゴリズムの解読の困難さと同じと考えられていた。しかし、ICカードが暗号処理を行っている時の消費電流を観測し、解析することにより、暗号アルゴリズムの解読より容易に暗号処理の内容や暗号鍵が推定される可能性が示唆されている。消費電流は、リーダライタから供給されている電流を測定することにより観測することができる。このような危険性は、John Wiley & sons社 W.Rankl & W.Effing著 「Smart CardHandbook」の8.5.1.1 Passive protective mechanisms(２６３ページ)に記載されている。

【０００７】ICカード用チップを構成しているCMOSは、出力状態が１から０あるいは０から１に変わった時に電流を消費する。特に、データバス２０３のバスは、大きな電気容量を持つため、バスの値が１から０あるいは０から１に変わると、大きな電流を消費する。そのため、消費電流を観測すれば、ICカード用チップの中で、何が動作しているか分かる可能性を示唆している。

【０００８】図３は、ICカード用チップの１サイクルでの消費電流の波形を示したものである。処理しているデータの依存して、電流波形が３０１や３０２のように異なる。このような差は、バス２０３を流れるデータや中央演算装置２０１で処理しているデータに依存して生じる。

【０００９】１６ビットのプリチャージバスにデータを転送する場合を考える。プリチャージバスは、データ転送の前にすべてのバスの値を“０”にそろえるバスである。このバスに、値は違うが“１”のビットの数が同じデータ、例えば、“１”のビットの数が２である１６進数で“８８”と“１１”、を転送した場合、電流波形はほぼ同じ波形になる。この理由は、“０”から“１”へ変化したビットの数が同じであるため、同じように電流をし消費し、同じ電流波形になったからである。もし、“１”のビットの数が１つ異なるデータ、例えば、“１”のビットの数が３である“８９”や“１９”を転

送した場合、“１”のビットの数が２のデータとは消費電流が異なる。これは、３ビット分バスの値が“０”から“１”に変わったため、その分の電流が消費される。そのため、先の２ビットが変化したデータに比べて消費電流が１ビット分大きくなる。一般に、“１”のビットの数が多いほど電流波形は高くなるという規則性がある。この規則性から、転送されているデータを推定することができる。

【００１０】図３の電流波形は、バスだけではなく、ICカード用チップを構成している構成要素の消費している電流の総和である。しかし、ICカード用チップのようなマイコンでは、主にバスにデータが転送されているフェーズ、主にCPUが演算をしているフェーズ、レジスタにデータを書き込んでいるフェーズなどに分かれている。そのため、フェーズに注目すれば、主にどの部分での消費電流の違いが分かり、その部分でのデータ操作を推定することができる。

【００１１】具体的な命令でどのように差が出るかを、次のような左シフト命令を例にして説明する。

【００１２】shiftl R1　　（式１）

この命令は、レジスタ１の内容を左シフトし、レジスタのビット列が左にシフトし、最上位ビットの値がキャリアとしてコンディションコードレジスタに入る命令である。レジスタＲ１の最上位ビットがデータバスを経由して、コンディションコードレジスタに転送されるため、電流波形の大きさを比較すれば、最上位のビットが“０”か“１”かが識別できる可能性がある。もしＲ１に重要なデータが入っていれば、そのデータの１ビットであるが、“０”か“１”かが分かる可能性がある。特に、DESのような暗号処理では、暗号鍵をシフトする操作が頻発する。このシフト操作のときに暗号鍵のデータを推定できる電流波形が生じ、暗号鍵を推定される危険性がある。

【００１３】これは、コ・プロセッサ２０２の演算でも同じである。演算内容が、暗号鍵に依存した偏りがあると、その偏りが消費電流から求められ、暗号鍵が推定される可能性がある。

【００１４】

【発明が解決しようとする課題】本発明の課題は、ICカード用チップでのデータ処理と消費電流との関連性を減らすことである。消費電流とチップの処理との関連性が減れば、観測した消費電流の波形からICカードチップ内での処理や暗号鍵の推測が困難になる。本発明の着眼点は、ICカードチップでの処理するデータを変形して処理することにより、消費電流の波形から、処理や暗号鍵の推測を困難にするものである。

【００１５】

【課題を解決するための手段】ICカード用チップに代表される耐タンパ装置は、プログラムを格納するプログラム格納部と、データを保存するデータ格納部を持つ記憶装置と、プログラムに従い、所定の処理を実行し、データ処理を行う中央演算装置を持ち、プログラムは、中央演算装置に実行の指示を与える処理命令から構成される一つ以上のデータ処理手段からなる、情報処理装置として考えることができる。本発明は、処理しているデータとICカード用チップの消費電流の関連性を減らすための方法として、処理するデータを撹乱用データで変形し、データの処理を変形したデータで処理し、処理後、撹乱用データを使って逆変形し、正しい処理結果を求めるものである。データ処理の後に使用する撹乱データは、必要があれば、データ処理と同じ処理を行った処理済の撹乱用データを使用する。また、撹乱用データは、データ処理ごとにランダムに設定する。このようにすることにより、データ処理において、本来のデータを使わず、毎回異なった変形したデータを使うことができる。そのため、電流波形からデータを推定することが困難にする。

【００１６】具体的には、まず、撹乱用データXiを作成し、データD1を変形し、変形データH1を作成する。変形の方法には、排他的論理和や加算、乗算などがある。データ処理では、変形データH1を使い、データ処理を行い、処理済み変形データH2を生成する。本来は、データD1を使用するのであるが、ここでは、変形データH1を使用しているため、変形データH1を処理する電流波形からデータD1のデータを推定することは困難である。変形データH1は、処理ごとに異なる撹乱用データXiにて、変形されているため、毎回、異なるデータとなっている。そのため、変形データH1を処理する電流波形も、毎回、異なった波形となり、その波形から変形データH1を推定しても意味がないことになる。

【００１７】撹乱用データXiもデータD1と同じようにデータ処理をする必要がある場合は、撹乱用データXiを入力とし、データ処理を行い、 処理済み撹乱用データXoを作成する。そして、処理済み撹乱用データXoを使って処理済み変形データH2を処理し、入力データD1を入力データ処理手段で処理した結果である処理済みデータD2を得る。

【００１８】データ変形の方法が異なったものを使用する必要がある場合などでは、データ変形を連結して、実行する必要がある場合がある。その場合は、データ変形、変形データ処理、撹乱用データ処理、データ逆変形処理を入れ子にしたり、組み合わせることにより、本来のデータをデータ処理しないようにデータ変形を連続させる。

【００１９】本発明は、暗号アルゴリズムでの、データを入れ替える転置や換字、あるいは、テーブルのアクセスなどの情報隠蔽に使うことができる。データの入れ替えの場合は、排他的論理和をデータ変形に使い、データ処理において、変形したデータと撹乱用データを同じように処理し、常に、変形したデータと撹乱用データを排他的論理和を実行すると、本来のデータを得ることがで

きるように変形処理を行うことが、有効な方法の一つである。

【００２０】

【発明の実施の形態】以下、本発明の実施例について図面を参照しながら説明する。

【００２１】図１はICカードの外観を示したものである。ICカード１０１は、ISO７８１６の規格により大きさや、ICカードのチップ１０２の位置や接点の数および割り当てなどが規定されている。

【００２２】図２は、ICカード用チップ１０２の内部構成である。構成については、従来技術の説明で既に述べたとおりである。本発明は、プログラム２０５で処理するデータに撹乱を加えることにより、処理中に生じるICカード用チップのハードウェアが消費する電流波形から、本物のデータの推定を困難にさせるものである。

【００２３】基本的な考えを次の簡単な命令列を例にして説明する。

【００２４】

```
logica_shiftl R1                    (式２)
xor R1 R2                           (式３)
```

式２は、レジスタＲ１の値を左に論理的にループさせる命令である。最上位ビットが最下位ビットに移動する。その結果はレジスタＲ１に格納される。その結果とレジスタＲ２の内容を排他的論理和し、その結果がレジスタＲ２に格納される命令列である。このような命令は、DESのような暗号アルゴリズムでは、頻繁に出現する。式２と式３では、処理するデータをそのまま扱っているのでデータの内容によって電流波形の大きさを変える。そのため、電流波形を観測することによりデータを推定できる可能性がある。

【００２５】式２と式３の命令で処理するデータを直接扱わないように、以下のような命令列に変更する。

【００２６】

```
xor X1 R1                           (式４)
xor X2 R2                           (式５)
logica_shiftl R1                    (式６)
xor R1 R2                           (式７)
logica_shiftl X1                    (式８)
xor X1 X2                           (式９)
xor X2 R2                           (式１０)
```

ここで、Ｘ１とＸ２は勝手に選んだ乱数であり、撹乱用データである。式４と式５では、レジスタＲ１とＲ２に乱数Ｘ１とＸ２を排他的論理和を行い、撹乱用データを使い、本物のデータの変形を行う変形処理である。式６と式７は式２と式３と同じデータ処理であるが、変形処理を行ったため、Ｒ１とＲ２の値は、本物のデータの値とは異なったデータとなっている。式８と式９は、撹乱用データ自体をデータ処理を行っているものである。式１０で、撹乱用データの処理結果と式７の処理結果を排他的論理和することにより、本物のデータにもどす逆変形処理である。これは、請求項７の実施例である。

【００２７】これを具体的な数値で値の変化を示すと以下のようになる。Ｒ１とＲ２を以下の値とする。

【００２８】

```
R1:11001010                         (式１１)
R2:01010111                         (式１２)
```

式２の結果のＲ１の値は以下のようになる。

【００２９】

```
R1:10010101                         (式１３)
```

そして、式３の処理結果は、以下のようになる。

【００３０】

```
R2:11000010                         (式１４)
```

次に、本発明の変形を行った場合について示す。まず、撹乱用データを以下のようにする。レジスタＲ１とＲ２の値は同じものとする。

【００３１】

```
X1:10010111                         (式１５)
X2:00111010                         (式１６)
```

式４と式５を処理した結果、レジスタＲ１とＲ２の値は以下のようになる。

【００３２】

```
R1:01011101                         (式１７)
R2:01101101                         (式１８)
```

式６と式７を処理した結果のレジスタＲ１とＲ２の値は以下のようになる。

【００３３】

```
R1:10111010                         (式１９)
R2:11010111                         (式２０)
```

式８と式９で撹乱用データＸ１、Ｘ２も同じようにデータ処理した結果は、以下のようになる。

【００３４】

```
X1:00101111                         (式２１)
X2:00010101                         (式２２)
```

そして式１０の逆変形処理した結果、本物のデータをそのまま扱った処理結果式１４と同じ結果が得られる。

【００３５】

```
R2:11000010                         (式２３)
```

この例で示したように、変形処理を行い、変形したデータも撹乱用データも同じデータ処理を行い、その結果を使って逆変形することにより、元の値を求めることができる。そして、データ処理では、本物のデータは使われていないため、その電流波形を見ても処理している変形データは推定できるが、本物のデータを推定することは困難である。

【００３６】上で具体的な数値で示した例をより一般的な形式で示す。実際の処理を以下の通りとする。

【００３７】

```
Output(j) = f(Input(i))             (式２４)
```

これは、i個の入力Inputを入力し、処理ｆを行い、ｊ個の出力Outputを出力する処理である。式２と３の例では、Ｒ１とＲ２の二つの入力であり、出力は一つで、Ｒ２に格納されていた例である。式２４の処理中の電流波

InputX(i) = h(Input(i)、X(i))　　（式２５）

OutputX(j) = f(inputX(i))　　（式２６）

Xoutput(j) = f(X(i))　　（式２７）

Output(j) = g(OutputX(i)、Xoutput(i))　　（式２８）

式２５は、撹乱用データＸ（ｉ）を使って、入力データInput(i)を変形し、変形した入力データInputX(i)を作成する変形処理である。hは変形操作である。式２６は、変形した入力データを使いデータ処理を行うデータ処理である。式２７は撹乱用データを入力データと同じように処理する撹乱用データ処理である、式２８は、変形した入力データのデータ処理結果OutputX(j)と撹乱用データの処理結果Xoutput(j)を変形の逆変換を行う逆変形処理である。gは逆変形操作である。

【００３９】先の例では、式４と５が式２５の処理に対応し、変形操作ｈは排他的論理和である。式６と式７は式２６の変形された入力データのデータ処理に対応する。式８と式９は、式２７の撹乱用データのデータ処理

Output = Input(i) + Input(2) - Input(3)　　（式２９）

のような操作は、以下のような変形処理を行い、

InputX(1) = Input(1) + X(1)　　（式３０）

InputX(2) = Input(2) + X(2)　　（式３１）

InputX(3) = Input(3) + X(3)　　（式３２）

変形した入力データを処理することにより変形した入力データの処理結果を得ることができる。

【００４１】

OutputX = InputX(1) + InputX(2) - InputX(3)　　（式３３）

撹乱用データの同じようにデータ処理を行う。

【００４２】

Xoutput = X(1) + X(2) - X(3)　　（式３４）

そして逆変形処理を行う。

【００４３】

Output = g(Xoutput) = OutputX - Xoutput

= InputX(1) + InputX(2) - InputX(3) - (X(1) + X(2) - X(3))

= Input(1) + Input(2) - Input(3　　（式３５）

正しい出力データを求めることができる。これは、加減算の演算では、ある値を加算して演算し、最期の結果に加算した値を減算すれば、正しい結果が求まることからである。これは、請求項８と９の実施例である。

【００４４】データ処理ｆが乗除算であれば、変形操作ｈに乗算、あるいは除算をえらび、逆変形操作に除算、あるいは乗算を選ぶことにより、変形処理と逆変形処理を実現することができる。この理由は、加減算と同じで

Output = Input(1) + Input(2) - Input(3) mod N　　（式３６）

入力Input(i)を以下のように変形する。

【００４７】

InputX (i) = Input(i) + k(i) * N　　（式３７）

そして、変形した入力データを使い、加減剰余演算を行う。

【００４８】

OutputX = InputX(1) + InputX(2) - InputX(3) mod N　　（式３８）

この式は、式３７を使うことにより

形から処理されるデータを推定することを困難にするために、以下のように行う。

【００３８】

であり、式１０は、式２８の逆変形処理である。変形操作ｇは排他的論理和である。変形操作ｈと逆変形操作ｇを何に選ぶかはデータ処理ｆの特性で決まる。式２と３の処理では、排他的論理和が、変形操作ｈであり、逆変形操作ｇでもある。これは、シフト操作やXOR操作などは、排他的論理和を変形操作ｈとして選ぶことにより、逆変形操作ｇも排他的論理和を選ぶことができる。排他的論理和は、同じデータを排他的論理和を取ったならば、論理的ゼロになり、排他的論論理和を取った操作が消えるからである。

【００４０】データ処理ｆが加減算であれば、変形操作ｈに加算、あるいは減算をえらび、逆変形操作に減算、あるいは加算を選ぶことができる。例えば、

【００４５】データ処理ｆが剰余演算の加減算であれば、変形操作ｈに法Ｎの整数倍の加減算を選ぶことができる。例えば、以下のような加減剰余演算を考える。

【００４６】

【００４７】

【００４８】

OutputX = (Input(1) + k(1)*N) + (Input(2) + k(2)*N) - (Input(3) + k(3)*N
) mod N

= (Input(1) + Input(2) - Input(3)) + (k(1)*N + k(2)*N - k(3)*N)
mod N　（式３９）

と変形できる。そして、剰余演算の特徴である

0 = k * N mod N　（式４０）

を使うことにより、式３９の２番目の括弧内の値は０となり、式３９は、

OutputX = (Input(1) + Input(2) - Input(3)) mod N　（式４１）

となる。すなわち、変形した入力データの演算結果は、本来の演算結果と同じである。これは、剰余演算の特徴を使うことにより、撹乱用データのデータ処理と逆変形

Xoutput(i) = k(1) * N + k(2) * N - k(3) * N mod N　（式４２）

が、そもそも０になるため、撹乱用データのデータ処理と逆変形処理が不要であるためである。これは、請求項１２の実施例である。

【００４９】データ処理ｆが剰余演算の乗算であれば、

Output = Input(1) * Input(2) * Input(3) mod N　（式４３）

入力Input(i)を以下のように変形する。

InputX (i) = Input(i) + k(i) * N + 1　（式４４）

そして、変形した入力データを使い、乗算剰余演算を行う。

OutputX = InputX(1) * InputX(2) * InputX(3) mod N　（式４５）

この式は、式４４を使うことにより

OutputX = (Input(1) * (k(1)*N + 1)) + (Input(2) + (k(2)*N + 1) - (Input(
3) + (k(4)*N + 1)) mod N
= (Input(1) * Input(2) * Input(3)) *
((k(1)*N + 1) * (k(2)*N + 1) * (k(3)*N + 1)) mod N　（式４６）

と変形できる。そして、剰余演算の特徴である

0 = k * N mod N　（式４７）

を使うことにより、式４６は、

OutputX = (Input(1) * Input(2) * Input(3)) * (1 * 1 * 1) mod N
= Input(1) * Input(2) * Input(3) mod N　（式４８）

となる。すなわち、変形した入力データの演算結果は、本物の演算結果と同じである。これも、剰余演算の特徴を使うことにより撹乱データのデータ処理と逆変形処理が不要である例である。これは、請求項２２の実施例である。

【００５３】また、データ処理ｆが剰余演算の乗算であれば、変形操作ｈにある数Xと法Nに対して逆数であるYを使うこともできる。

Output = Input(1) * Input(2) * Input(3) mod N　（式５０）

入力Input(i)を以下のように変形する。

InputX (i) = Input(i) * X　（式５１）

そして、変形した入力データを使い、加減剰余演算を行う。

OutputX = InputX(1) * InputX(2) * InputX(3) mod N　（式５２）

そして、逆変形処理ｇとして、Xを掛けた回数だけYを掛けたものを選ぶと、OutputXに逆変形処理ｇを操作する

Output = OutputX * Y * Y * Y mod N

【００５０】

【００５１】

【００５２】

【００５４】

1 = X * Y mod N　（式４９）

このような数の簡単なものとして、Ｘが２で、Ｙが（N+1）/2であるものがある。これは、変形操作としてＸを掛けておき、逆変形でＸを掛けた回数だけＹを掛けることにより、本物の処理結果を求めるものである。例えば、以下のような加減剰余演算を考える。

【００５５】

【００５６】

【００５７】

【００５８】

= InputX(1) * InputX(2) * InputX(3) * Y * Y * Y mod N
= Input(1) * X * Input(2) * X * Input(3) * X * Y * Y * Y mod N
= Input(1) * Input(2) * Input(3) * X * X * X* Y * Y * Y mod N
= Input(1) * Input(2) * Input(3) * X * Y * X * Y * X * Y mod N
= Input(1) * Input(2) * Input(3) mod N　　（式５３）

式５３では、式４９の性質を使った。この例では、撹乱用データのデータ処理は不要であるが、撹乱用データを掛けた回数だけ、法Ｎでその逆数を逆変形処理で掛ければ、正しい結果を得ることができる。これは、請求項１３と１４の実施例である。

【００５９】これまでは、データ処理ｆがデータ操作である例であったが、テーブルからのデータ取り出し操作を撹乱することも電流波形からデータを推定させないために必要である。テーブルのデータの撹乱とテーブルのアドレスの撹乱に関する実施例について、図３３のテーブルからデータを取り出す例を用いて説明する。

【００６０】図３３のテーブルのデータを撹乱用データX1で排他的論理和をとる。ここで撹乱用データとして“９”を選び、テーブルの値と９との排他的論理和を実行する。その結果が図３４のテーブルである。次に、テーブルのアドレスを撹乱するために、行番号と撹乱用データX2として選んだ“３”との排他的論理和を実行し、

Gyou = GyouY1 xor Y1
Retsu = RetsuY2 xor Y2

しかし、この逆変形処理を行って図３３のテーブルを使うことは、真のアドレスデータを使うため、電流波形からアドレスデータを推定される可能性がある。そこで、まず、図３５のテーブルを作った時に、行番号と列番号

GyouY1X2 = GyouY1 xor X2
RetsuY1X3 = RetsuY2 xor X3

そして、今まで使ってきた撹乱用データの逆変形処理を行う。

GyouX2 = GyouY1X2 xor Y1
RetsuX3 = RetsuY2X3 xor Y2

こうすることにより、真の行番号や列番号が使われていないため、電流波形からその値を推定することは困難である。GyouX2とRetsuX3を使って、図３５のテーブルTab

DataX1 = TableX1X2X3(GyouX2、RetsuX3)　　（式６０）

図３５のテーブルはすでに撹乱用データX1で変形されているため、これ以降は、撹乱用データをX1として処理を行う。式５６から式６０までの処理で真のデータを使用していない。これは、請求項１６と１７の実施例である。

【００６６】一定の処理ごとに、テーブルのデータの撹乱用データX1や行番号の撹乱用データX2、列番号の撹乱用データX3をランダムに生成し、テーブルを変形しておく。こうすることにより、処理ごとにテーブルが変形されるため、電流波形からデータを推定することは困難である。

【００６７】以上撹乱用データの種類とデータの変形の列番号と撹乱用データX3として選んだ“２”との排他的論理和を実行し、テーブルを並び替える。その結果が図３５である。元のテーブルである図３３のテーブルの１行２列目のデータ３３０１である“０”は、撹乱用データX1と排他的論理和を実行した結果、図３４のテーブルでは“９”（３４０１）となる。そして、行番号と列番号を撹乱用データX2と撹乱用データX3とそれぞれ排他的論理和を実行した結果、３４０１は、３５０１の場所に移る。テーブルからのデータ取り出し操作を撹乱するために、このようなテーブルを作成しておく。

【００６１】アドレス計算までに、すでに、撹乱用データY1とY2でそれぞれ、行番号変数Gyouと列番号変数Retsuが排他的論理和で変形されているとする。すなわち、正しい行番号Gyouと列番号RetsuはY1とY2をGyouとRetsuに排他的論理和を実行しないと得られないとする。それは、以下のような関係である。

【００６２】
（式５４）
（式５５）

を撹乱するために使用した撹乱用データX2と撹乱用データX3を用いる。

【００６３】

（式５６）
（式５７）

【００６４】

（式５８）
（式５９）

leX1X2X3を参照しDataX1を取る出す。

【００６５】

方法について説明してきた。次に、その処理手順を示す。図４に基本的な撹乱用データを使った情報隠蔽手順の実施例について示す。

【００６８】図４は、基本的な手順である。撹乱用データ生成手段で撹乱用データXiを生成する（４０１）。この一般的な方法としては、乱数発生器や擬似乱数を使って必要な長さの乱数を生成する方法がある。次に、データ変形処理手段（４０６）で、入力データD1を撹乱用データXi（４０５）で変形し、変形データH1（４０７）を生成する。変形方法は、先に述べたように、排他的論理和や、加減算、乗除算などがある。そして変形データＨ１を使ってデータ処理を変形データ処理手段（４０８）

で行い、処理済み変形データＨ２を生成する。一方、撹乱用データＸｉは入力データと同じデータ処理を撹乱用データ処理手段(４０３)で行い、処理済みの撹乱用データXo（４０４）を作成する。そして、撹乱用データXoと処理済み変形データＨ２を使い、データ逆変形処理手段（４１０）で真の処理済みデータD2（４１１）を求める。データ変形処理手段（４０６）やデータ逆変形処理手段（４１０）の方法には、すでに説明した、排他的論理和や加減算、乗除算、剰余演算などがある。これは、請求項１の実施例である。

【００６９】図５の実施例は、撹乱用データを二つ使った例であり、第１の撹乱用データを使った情報隠蔽処理の中に第２の撹乱用データを使った情報隠蔽処理が含まれている場合である。主な流れは、図４の実施例と同じ

```
H1 = D1 xor X1i
H2 = f1(H1)
X1o = f1(X1i)
H31 = H2 xor X2i
H32 = D2 xor X2i
H4 = f2(H31、H32)
X2o = f2(X2i、X2i)
H5 = H4 xor X2o
D2 = H5 xor X1o
```

ここでｆ１とｆ２はデータ処理操作である。この例のように、第２のデータ処理ｆ２で、別のデータＤ２を使う場合で、そのデータＤ２を第２の撹乱用データで変形して使用する場合に、本実施例の処理手順を使うと有効である。これは、請求項２の実施例である。

【００７１】図６の実施例も、撹乱用データを二つ使った例である。図５の実施例との大きな違いは、第１の撹乱用データを使った情報隠蔽処理と第２の撹乱用データを使った情報隠蔽処理がつながっている場合である。そして、第１の撹乱用データによる変換を戻す逆変形処理の前に、第２の撹乱用データで変形しておき、真のデータを使った処理を隠蔽するための手順である。第１の撹乱用データで変形された変形データＨ１（６０９）は、

```
H1 = D1 xor X1o
H2 = f1(H1)
X1o = f1(X1i)
H3 = H2 xor X2i
H4 = H3 xor X1o
H5 = f2(H4)
X2o = f2(X2i)
D2 = H5 xor X2i
```

これは、処理操作が複数あり、撹乱用データを複数使う必要な場合に有効である。これは、請求項３の実施例である。

【００７３】図７の実施例は、撹乱用データの面からは、データ処理をあらかじめ計算しておき、処理を効率化するものである。あらかじめ、撹乱用データ処理手段である。第１の撹乱用データで変形された変形データＨ１（５０７）をデータ処理し、その処理結果である処理済み変形データＨ２（５０９）に第２の撹乱用データX2iを使って第２データ変形処理手段（５１０）で変形を行い、処理済み変形データＨ３(５１１)を作成する。そのデータを第２変形データ処理手段（５１２）でデータ処理を行い、処理済み変形データＨ４を生成し、第２の撹乱用データの逆変換を第２データ逆変形処理手段（５２０）でおこない、処理済み変形データＨ５（５２１）を生成する。そして、第１の撹乱用データの逆変換を第１データ逆変形処理手段（５１４）で行い、真の処理済みデータＤ２（５１５）を求める。変形に排他的論理和を使った例は、以下の通りである。

【００７０】

（式６１）

第１変形データ処理手段（６１０）でデータ処理を行い、その処理済み変形データＨ２（６１１）を第２変形データ作成手段（６１２）で第２の撹乱用データを使って変形し、処理済み変形データＨ３(６１３)を生成する。そして、第１撹乱用データの逆変換を第１データ逆変形処理手段（６０５）で行う。その処理結果である処理済み変形データＨ４（６０６）を使って第２変形データ処理手段（６１４）を行い、処理済み変形データＨ５(６１５)を生成する。そして、第２データ逆変形処理手段(６１６)で、逆変換を行い、真の処理済みデータＤ２(６１７)を生成する。変形に排他的論理和を使った例は、以下の通りである。

【００７２】

（式６２）

で処理済み撹乱用データXoを生成し（７０３）、処理済み撹乱用データ記録手段（７０６）で、記録しておく。処理の中では、データ逆変形処理手段（７１３）では、記録しておいた処理済み撹乱用データ（７１４）を読みだし、使用する。これは、同じようなデータ処理を何回も実行する場合には効率の面で有効である。しかし、撹

乱用データが何回も使われるため、情報の隠蔽性にとって、図４の実施例のように撹乱用データを毎回変更する方が、有効である。これは、処理速度と情報の隠蔽性のトレードオフから決まる。これは、請求項５の実施例である。

【００７４】図８の実施例は、第１と第２の撹乱用データの逆変形を統合して行い、その後その結果を使って、データの逆変形を行うものである。第１と第２の撹乱用データは、それぞれ、第１撹乱用データ処理手段（８０３）と第２撹乱用データ処理手段（８０７）で処理済み撹乱用データX1oとX2oが作られる。それらのデータをデータ逆変形統合手段で統合し、統合処理済み撹乱用データXoを生成する。そのデータを使い、第１変形データ処理(８１４)及び第２変形データ処理(８１８)を処理した結果である処理済み変形データＨ４(８１９)の逆変形処理手段（８２０）を行い、真の処理済みデータＤ２を生成する。これは、個別に逆変形する場合よりも、処理済み撹乱用データを統合し、そのあと、まとめて変形処理を行う方法である。逆変形処理に処理時間がかかる場合に有効である。これは、請求項４の実施例である。

【００７５】次に、対称暗号DES(data encryption standard)を例にした実施例について説明する。本発明は、他の暗号にも使うことができる。暗号については、共立出版株式会社岡本栄司著「暗号理論入門」などに、記載されている。

【００７６】DESは、６４ビットのデータ（平文か暗号文）を５６ビットの暗号鍵で、暗号化と復号化を行う。暗号化と復号化で同じ暗号鍵を使用するため、対称暗号と呼ばれている。DESは、トランプをきって、ランダムにするように、平分（暗号化される文）のビットをランダムに入れ替え暗号化する。データの入れ替えは、暗号鍵にしたがって行う。復号化する時は、暗号鍵に従い入れ替えた順と逆に入れ替えを行い、データを元に戻す。DESの処理での入れ替えは、ビット単位と複数ビットまとまった単位での二つの入れ替え方法がある。前者は、転置と呼び、後者は、換字と呼ぶ。

【００７７】図９を用いて、DESの暗号化の方法を説明する。変形処理ａ（９０１）と変形処理ｂ(９０４)、逆変形処理(９１６)は、本発明に関するものであり、DESの本来の暗号処理には関係がない。暗号文は、まず、初期転置(IP：Initisal Permutation)９０２で転置させられる。これは、初期転置テーブルに従い、ビット単位で、暗号文の６４ビットのデータを入れ替える。これ以降、初期転置の逆転置（IP-1）９１６まで、一組の操作を１６段行う。

【００７８】各１段の処理は、前段の結果の前半か後半の３２ビットのデータと暗号鍵を入力としてｆ関数９０３と呼ばれる処理を行い、その出力を前段の残りの半分のビットを使って排他的論理和９０９を取る操作を行う。暗号鍵も、入れ替えが行われる。暗号鍵に対して、まず、ＰＣ－１というテーブルを使った選択転置PC-1（９０５）が行われる。その後、ＰＣ－２というテーブルを使った選択転置ＰＣ－２（９０８）を行い、暗号鍵の入れ替えを行う。次の段では、２８ビットづつをＬＳテーブルにしたがって巡回させて使用する。

【００７９】本実施例では、IP処理の前に、平文を変形するための変形処理ａ（９０１）と暗号鍵を変形するための変形処理ｂ(９０４)、そして、最期に逆変形する処理(９１６)を追加する。変形処理ａ（９０１）は、平文を変形し、IP処理（９０２）やｆ関数（９０３）の処理で、平文そのものを処理するのではなく、変形した平文を処理することにより、その処理の電流波形から平文のデータを推定できないようにするものである。変形処理ｂ（９０４）は、PC-1処理(９０５)やＬＳ処理(９０７)、ＰＣ－２処理(９０８)、ｆ関数（９０３）で暗号鍵そのものを処理するのではなく変形した暗号鍵を処理することにより、その処理の電流波形から暗号鍵の推定を困難にするものである。

【００８０】ｆ関数３０３の処理を、図１０に示す。まず、ｆ関数への入力文を選択的転置行列Eに基づいて、選択的転置を行う（１００２）。次に、選択的転置を行った入力データと暗号鍵との排他的論理和をとり（１００３）、Ｓボックスの処理を行い（１００４）、Ｐ転置処理を行う（１００５）。Ｓボックスの処理は、１００３の排他的論理和の処理結果である４８ビットから６ビットづつ取り出し、８つのＳボックステーブルの行番号と列番号を求め、４ビットのデータを生成する処理である。各Ｓボックステーブルは、６ビットごとで示されているデータの位置により異なる。Ｐ転置処理は、Ｐ転置テーブルに従い、３２ビットのビット位置を入れ替える操作である。

【００８１】変形処理ａ（９０１）と変形処理ｂ（９０２）は基本的に同じ処理である。図１１を用いて変形処理ａの平文の変形データの作成処理について説明する。撹乱用データＸ１をランダムに生成する。これは、DESの暗号化（あるいは復号化）処理ごとに乱数発生器か擬似乱数を使って生成する（１１０２）。毎回、異なった撹乱用データを使用する。次に撹乱用データＸ１と平文PをXOR(排他的論理和)を実行し、変形平文（変形した平文)PX1を生成する（１１０３）。DESの場合は、平文は６４ビットであるが、生成する乱数は、６４ビットでも８ビットでも構わない。しかし、６４ビット以下ならば、拡張するようなことを行い、６４ビットの撹乱用データＸ１を生成する必要がある。生成した乱数が８ビットであれば、それを８回繰りかえして６４ビットの撹乱用データＸ１を生成してもよい。ここで、排他的論理和（XOR）を使って変形したため、撹乱用データＸ１と変形平文PX1をXORすると、平文Pが生成される。これは、請求項６の実施例でもある。

【００８２】変形処理ｂ（９０４）の暗号鍵の変形デー

タの作成手順を図３６に示す通りである。平文と撹乱用データＸ１の代わりに暗号鍵Kと撹乱用データＸ２を用いている点が、図１１の実施例と異なるだけである。DESでは、暗号鍵は平文と同じ６４ビットである。処理で変形された暗号鍵KX2が生成される。

【００８３】次に、IP処理（９０２）について説明する。IP処理は、図３７に示すテーブルにしたがって平文６４ビットの並びを入れ替えるものである。テーブルに従い、出力の第１ビットは入力の第５８ビット、出力の第２ビットは入力の第５０ビット、出力の第６４ビットは入力の第７ビットと入れ替える。本実施例でのIP処理を、図１２を用いて説明する。まず、変形平文PX1をIP処理し、IP処理済み変形平文PX1IPを生成する（１２０２）。ビットの入れ替えは、図３７のテーブルに従う。次に、撹乱用データＸ１も同じようにIP処理し、IP処理済み撹乱用データX1IPを生成する（１２０３）。IP処理済み変形平文PX1IPとIP処理済み撹乱用データX1IPとは排他的論理和を取ると、平文をIP処理した結果を生成することができる。これは、IP処理がビットの移動であるため、撹乱用データも変形平文ＰＸ１と同じように移動したため、ビットごとに排他的論理和を実行すると真のデータが求まるという関係が維持されているためである。IP処理の下位３２ビットが第１段のｆ関数（９０３）と第２段の排他的論理和に使われ、上位３２ビットは、排他的論理和（９０９）の入力となる。これは、請求項１、７、１８、２０の実施例である。

【００８４】IP処理では、変形平文ＰＸ１のビットの値は、元の平文のビットの値と異なるため、IP処理の電流波形を見ても、平文のデータを推定することは困難である。“１”のビットの数によって、消費電流は大きくなるが、変形平文での“１”のビットの数は、平文の“１”のビットの数と何ら関係はない。そのため、電流波形の大きさから、平文のデータを推定することは困難である。このように、平文を撹乱用データで変形することにより、処理中の電流波形を観測しても元のデータを推定することが困難にすることができる。

【００８５】PC-1処理は、IP処理とほとんど同じである。図３８のPC-1用の変換テーブルを使い、６４ビットの暗号鍵をパリティビットの８ビットを取り除き、５６ビットにし、かつビットの順序を入れ替えるものである。図３８のテーブルの見方は図３７のテーブルの見かたと同じである。ＰＣ－１処理済みの変形暗号鍵KX2PC1とPC-1処理済みの撹乱用データX2PC1は、排他的論理和を実行すると、正しいPC-1処理済みの暗号鍵が求まる。

【００８６】LS処理は、PC-1処理で生成された５８ビットの鍵を右と左に２８ビットずつに分け、それぞれＬＳテーブルにしたがって左に１ビットあるいは２ビットシフトするものである。実施例を図１５を用いて説明する。まず、１５０２で、PC-1処理処理済み変形暗号鍵KX2PC1した結果をＬＳ処理し、PC-1及びLS処理済み変形暗号鍵KX2PC1LSを生成し、１５０３で、PC-1処理処理済み撹乱用データX2PC1した結果をＬＳ処理し、PC-1及びLS処理済み撹乱用データX2PC1LSを生成する。LS処理もビット位置の入れ替えであるため、PC-1及びLS処理済み変形暗号鍵KX2PC1LSとPC-1及びLS処理済み撹乱用データX2PC1LSの排他的論理和を実行すると、真のLSを処理した暗号鍵の結果が得られる。LS処理でも、撹乱用データを使ったため、実際に操作されているデータは、真の暗号鍵とは違うため、電流波形を観測しても、暗号鍵の推定は困難である。

【００８７】PC-2処理は、LSの処理結果の５６ビットをPC-2テーブルにしたがって、４８ビットに縮約型転置する。１４０２で、PC-1及びLS処理済み変形暗号鍵KX2PC1LSをPC-2処理し、PC-1及びLS、PC-2処理済み変形暗号鍵KX2PC1LSPC2を生成する。１４０３で、PC-1及びLS処理済み撹乱用データX2PC1LSをPC-2処理し、PC-1及びLS、PC-2処理済み撹乱用データX2PC1LSPC2を生成する。基本的には、テーブルを使った転置であるため、PC-1処理と基本的に同じである。

【００８８】次に、ｆ関数９０３の処理について説明する。ｆ関数は、図１０に示すように、選択的転置E処理（１００２）、暗号鍵と選択的転置の実行結果の排他的論理和(１００３)、Ｓボックス処理(１００４)、Ｐ転置処理(１００５)からなる。

【００８９】選択的転置E処理について図１６を用いて説明する。選択的転置Eは、IP処理と同じように図２８の転置テーブルを使って、ビットの並びを変更するものである。１６０２で、IP処理済み変形平文PXIPを選択的転置E処理し、IP処理及びE転置処理済み変形平文PXIPEを生成する。そして、１６０３でIP処理済み撹乱用データXIPを選択的転置E処理し、IP処理及びE転置処理済み撹乱用データXIPEを生成する。IP処理やPC-1処理と同じように、IP処理及びE転置処理済み変形平文PXIPEとIP処理及びE転置処理済み撹乱用データXIPEを排他的論理和をとれば、正しいIP処理及びE転置処理済み平文が求まる。また、転置Eテーブルを使ってビットに入れかを行う際も、ビットの値は、変形されているビットの値であるので、その処理の電流波形を見ても真のデータを推定することは困難である。

【００９０】次に、ｆ関数での２番目の処理である暗号鍵と選択的転置の実行結果の排他的論理和の処理を行う。この処理について、図１７を用いて説明する。１７０２では、平文から生成したIP処理及びE転置処理済み変形平文PXIPEと、暗号鍵から生成したPC-1及びLS、PC-2処理済み変形暗号鍵KX2PC1LSPC2とをXORし、Ｓボックス処理の入力となる４８ビットのＳボックス入力データSinputXを作成する。次に１７０３では、平文用の撹乱用データから生成したIP処理及びE転置処理済み撹乱用データXIPEと、暗号鍵用の撹乱用データから生成したPC-1及びLS、PC-2処理済み撹乱用データX2PC1LSPC2とをXO

Rし、Ｓボックス入力データSinputX用の撹乱データであるＳボックス入力データ撹乱用データXSinputを生成する。排他的論理和の性質から、Ｓボックス入力データ撹乱用データXSinputは、二つの撹乱用データ（PC-1及びLS、PC-2処理済み撹乱用データX2PC1LSPC2と、PC-1及びLS、PC-2処理済み撹乱用データX2PC1LSPC2）をXORするこ

PX1 = P xor X1

KX2 = K xor X2

そうすると、PとKの排他的論理和を実行した結果をＺとすると、PX1とPX2の排他的論理和を実行した結果Ｚ１と

Z = P xor K

Z1 = PX1 xor KX2

= (P xor X1) xor (K xor X2)

= P xor X1 xor K xro X2

= (P xor K) xor (X1 xor X2)

= Z xor (X1 xor X2)

すなわち、Ｚ１を真のデータに戻すための撹乱用データとしてＰとＫの撹乱用データを排他的論理和を使えばよいことが分かる。暗号鍵と選択的転置の実行結果の排他的論理和の処理では、Ｓボックス入力データ SinputXの撹乱用データとして、平文用の撹乱用データから生成したIP処理及びE転置処理済み撹乱用データXIPEと、暗号鍵用の撹乱用データから生成したPC-1及びLS、PC-2処理済み撹乱用データX2PC1LSPC2とをXORして生成したＳボックス入力データ撹乱用データXSinputを使えばよいことが分かる。これは、請求項１８の実施例である。

【００９３】次に、Ｓボックスの処理について図１８を用いて説明する。Ｓボックス入力データSinputXから６ビットずつ取り出し、８つのＳボックスについて処理を行う。DESで使われている１番目のＳボックスを、図２５で示す。８つのＳボックスは形式は同じであるが、それぞれフィールドのデータが異なる。それぞれのＳボックスの処理では、まず、Ｓボックス入力データSinputの上位i番目の６ビットのサブデータSubSinputX(i)を取り出す（１８０５）。前もって作成しておいたＳボックスを変形した変形Ｓボックステーブルのアドレス撹乱用データXsa(i)とSubSinputX(i)をXORし、SubSinputXXsa(i)を生成する（１８０６）。そのSubSinputXsa(i)をＳボックス入力データ撹乱用データXSinputの上位i番目の６ビットとXORし、SubSinputXsa(i)を生成する（１８０７）。SubSinputXsa(i)は、i番目のＳボックスを取り出すための本物のアドレスデータに対して、アドレス撹乱用データXsa(i)が排他的論理和された値である。SubSinputX(i)とXSinput(i)を排他的論理和（XOR）すると、それは、真のデータに戻るのであるので、XSinput(i)と排他的論理和する前に、SubSinputX(i)とXsa(i)の排他的論理和を行い、その後で、XSinput(i)を排他的論理和（XOR）を実行する。こうすることにより、本物のデータの処理を行う必要がなくなり、その電流波形からデータを推定することが困難になっている。次に、SubSinputにより生成できる。これを簡単な例で示す。ここ、平文をPと鍵をK、その変形平をPX1、変形鍵をKX2とする。それらの関係は、式６３と式６４である。X1とX2は、平文と鍵の撹乱用データである。

【００９１】

（式６３）

（式６４）

Ｚの関係は、以下のようになる。

【００９２】

（式６５）

（式６６）

tXsa(i)を使って、変形Ｓボックステーブルのアドレスを計算する（１８０８）。本来のＳボックスのテーブルを参照するためのアドレスが変形されているため、テーブルをあらかじめ変形しておく必要がある。計算したアドレスを元に、変形ＳボックステーブルＳ（ｉ）からＳボックス出力データSoutputX3(i)を取り出す（１８０９）。そして、同時にＳボックスの出力データSoutputX3(i)用の撹乱用データX3(i)を取り出す（１８１０）。８つのＳボックスを処理し、 SoutputX3(i)とX3(i)をiを１から８までのデータをそれぞれ連結することによって、SoutputX3とX3を生成する。これ以降は、処理データは、SoutputX3となり、撹乱用データはX3となる。これは、請求項２１の実施例である。

【００９４】次に、変形Ｓボックステーブルの生成方法について図２３と図２４を用いて説明する。ここのＳボックスで、Ｓ（ｉ）ボックス用のアドレス撹乱用データXsa(i)とデータ撹乱用データX3(i)を作成する（２３０６）。Xsa(i)は６ビットであり、X3(i)は４ビットである。撹乱用データX3は４ビットずつ作成したX3(i)を８個集めた３２ビットのデータである。次に、変形Ｓ(ｉ)ボックステーブル作成ルーチンを呼びだす（２３０７）。i番目の変形Ｓボックステーブル作成ルーチンの処理を図２４で説明する。kは行番を指定し、１は列番号を指定する。ｋ行１列の処理は２４０８から２４１３である。１番目のＳボックスのテーブルは、図２５に示す通りである。まず、i番目のオリジナルのＳボックスの行番号ｋ列番号１のデータdを取り出す（２４０８）。そして、そのデータｄと撹乱用データX3(i)を排他的論理和を取り、それをｄ２とする（２４０９）。これは、撹乱用データを“7”とすると、元のＳボックスのデータ２５０４に対して、その結果は２６０４になる。これをすべてのフィールドに行うと、図２６のようになる。図２６のテーブルは、図２５の１番目のＳボックスのデータに対して、撹乱用データを“7”として排

他的論理和を取ったものである。

【００９５】次に、アドレスの撹乱を行う。まず、Xsa1をXsa(i)の上位１ビットと下位１ビットから作った２ビットとし、Xsa2を Xsa(i)の上位２ビットから５ビットから作った４ビットのデータとする。これは、Ｓボックスのアドレスの計算方法に由来している。そして、図２６の行番号と列番号をそれぞれｋと１とすると、それぞれにXsa1とXsa2を排他的論理和を行う（２４１２）。新しくできた行番号と列番号をｋ２と１２とすると、i番目の変形Ｓボックステーブル Ｓ（ｉ）のk2行12列にデータd2を格納する（２４１３）。この処理の例を図２７に示す。図２７は、図２６の行と列に対して撹乱用データとして“2”と“9”を選んで作成したものである。わかりやすいように、行と列のデータの位置はそのままにして行と列の番号だけを変更している。図２５で、３行１列のデータ１２（２５０４）は、図２７では、１行８列に移り、値が１１に変形されている。この例では、データの撹乱用データは“7”であり、アドレス用の撹乱データは行が“2”であり、列が“9”である。このようにして８つのＳボックスを変形する。本実施例では、この処理は、DESの最初に行っておく。変形したＳボックステーブルは、DESの１６段で使用する。これは、請求項５の実施例でもある。

【００９６】Ｓボックスの処理が終了すると、処理データは、３２ビットのSoutputX3となり、撹乱用データは３２ビットのX3となる。これが、ｆ関数の最後の処理である転置Ｐ処理（１００５）の入力となる。転置Ｐ処理を図１９を用いて説明する。Ｓボックスの出力であるSinputX3を転置Ｐし、SinputX3Pを生成する（１９０２）。SinputX3の撹乱用データX3を転置Ｐし、X3Pを生成する（１９０３）。転地Ｐに使用するテーブルは図２９の通りである。このテーブルの使い方もIP処理のテーブルと同じである。

【００９７】ｆ関数の処理が終わると、転置Ｐ処理と前段の結果のXORを行う（９０９とか９１４）である。これは、まず、Ｓボックスの結果を転置PしたSinputX3Pと前段の結果をXORする（２００２）。そして、X3Pと前段の撹乱用データＸをXORする（２００３）。このXORの処理は、選択的転置E処理結果と暗号鍵のXOR処理（１７０１）と同じである。

【００９８】DESでは、最後にIP-1処理（９１５）を行う。この処理を図２１に示す。IP-1処理は、IP処理と似たようなビットの並びの入れ替え処理であり、IPテーブルの代わりにIP-1テーブルを使用する（２１０２）。これまでの処理結果をIP-1処理し、同様に撹乱用データＸをIP-1処理する（２１０３）。

【００９９】最後に、正しい処理結果に戻すために、逆変形処理を行う（９１６）。逆変形処理を図２２に示す。IP-1処理結果をIP-1処理した撹乱用データＸでXORすることにより、正しい結果を求める。ここで、初めて、変形されていない正しい処理結果が得られる。

【０１００】これまで処理されるデータの隠蔽について述べてきたが,撹乱用データも隠蔽する必要がある場合がある。基本的考え方は,撹乱用データを撹乱用の撹乱用データXRと排他的論理和を取り,変形することである。ただし,XRは,固定にしておき,あらかじめ,ビット位置の入れ替えなどを計算し逆変形のためのXRoを求めておく。そして,撹乱用データが必要になったときに,XRoを使って,元の撹乱用データを求めることにより,撹乱用データの変形と逆変形の実施と効率化を図る。まず,暗号鍵用の撹乱用データを例にして説明する。図３０の処理は,暗号鍵用の撹乱用データを撹乱用の撹乱用データXRで排他的論理和をとる変形操作である。変形処理ｂ（３６０１）で撹乱用データＸ２を生成した後に,図３０の撹乱用データの変形処理を行う。暗号鍵用の撹乱用データＸ２はＰＣ―１処理やLS処理,PC-2処理を受ける。これらは,あらかじめ,決まったビットの位置の入れ替えであるので,前もって決まっている値XRに対して,PC-2処理まで実行したときの撹乱用データXRoを求めておき記録しておく（３１０２から３１０５）。PC-2処理した後に,１４０３で生成したPC-1及びLS,PC-2処理済み撹乱用データX2PC1LSPC2に記録しておいた撹乱用データXRoをXORして（３２０２）,本来のPC-1及びLS,PC-2処理済み撹乱用データX2PC1LSPC2を求めることができる。このようにすることにより,撹乱用データも隠蔽することができる。ここで,撹乱用の撹乱用データXRとその処理済撹乱ｈ尿データXRoは,同じデータを使ってもよい。

【０１０１】DESでの実施例は,暗号化であるが,DESのアルゴリズムは,復号化でもほとんど同じであるため,本実施例はほとんど変更しないで適用できる。また,DES以外の暗号アルゴリズムも,転置処理や換字処理,剰余演算を多く使っているため,本発明を適用し,データを変形し,電流波形から本来のデータを推定することを困難にすることができる。

【０１０２】

【発明の効果】本発明によれば,ICカードチップでの処理データを変形することにより,消費電流の波形から,処理や暗号鍵の推測を困難になる。

【図面の簡単な説明】

【図１】ICカードのハードウェア構成。

【図２】ICカード用チップ内のハードウェア構成。

【図３】消費電流の波形。

【図４】一つの撹乱用データを使ったデータ変形の手順。

【図５】入れ子で二つの撹乱用データを使ったデータ変形の手順。

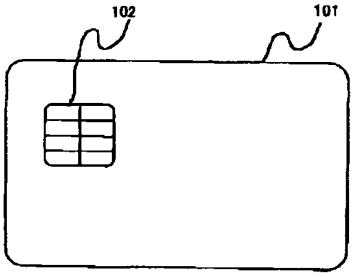
【図６】連続で二つの撹乱用データを使ったデータ変形の手順。

【図７】撹乱用データのデータ処理をあらかじめ計算しておくデータ変形の手順。

【図８】二つの撹乱用データの逆変形を統合したデータ変形の手順。
【図９】DESの全体処理の流れ。
【図１０】DESのｆ関数の処理の流れ。
【図１１】変形処理ａ。
【図１２】IP処理。
【図１３】PC-1処理。
【図１４】PC-2処理。
【図１５】ＬＳ処理。
【図１６】選択的転置E処理。
【図１７】選択的転置E処理結果と暗号鍵のXOR処理。
【図１８】Ｓボックス処理。
【図１９】転置Ｐ処理。
【図２０】転置Ｐ処理と前段の結果のXOR処理。
【図２１】IP-1処理。
【図２２】逆変形処理。
【図２３】変形Ｓボックスのテーブルの作成。
【図２４】i番目の変形Ｓボックステーブル作成ルーチン。
【図２５】１番目のＳボックステーブル。
【図２６】１番目のＳボックステーブルのデータを変形したテーブル。
【図２７】１番目のＳボックステーブルの配置を変形したテーブル。
【図２８】選択的転置Eテーブル。
【図２９】転置Ｐテーブル。
【図３０】撹乱用データの暗号化処理。
【図３１】撹乱用データの暗号化用データの変形計算処理。
【図３２】撹乱用データの復号化処理
【図３３】オリジナルテーブルの例。
【図３４】図３３の内容を変形したテーブル。
【図３５】図３４のテーブルの配置を変形したテーブル。
【図３６】変形処理ｂ。
【図３７】IP転置テーブル。
【図３８】PC-1選択的転置テーブル。

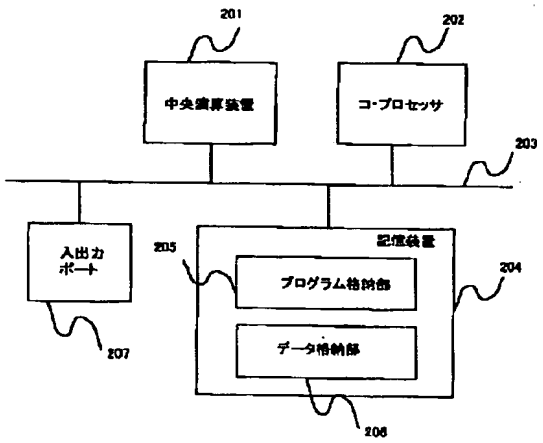
【図１】

図1



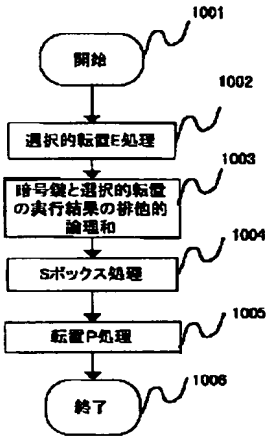

【図２】

図2



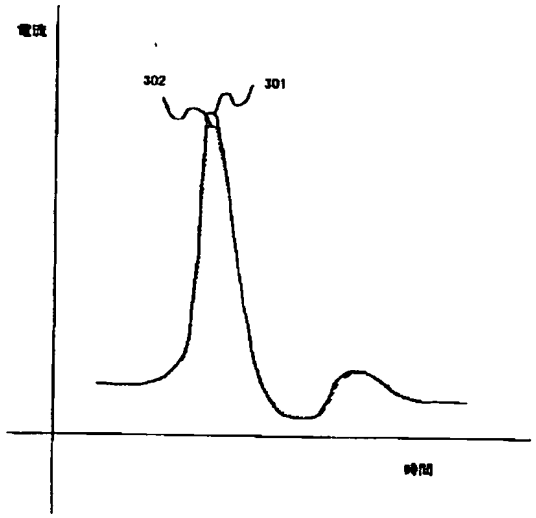

【図１０】

図10



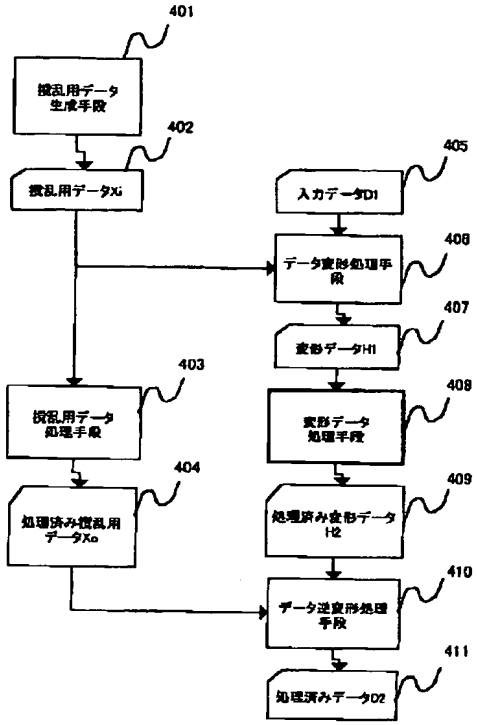
【図３】

図3



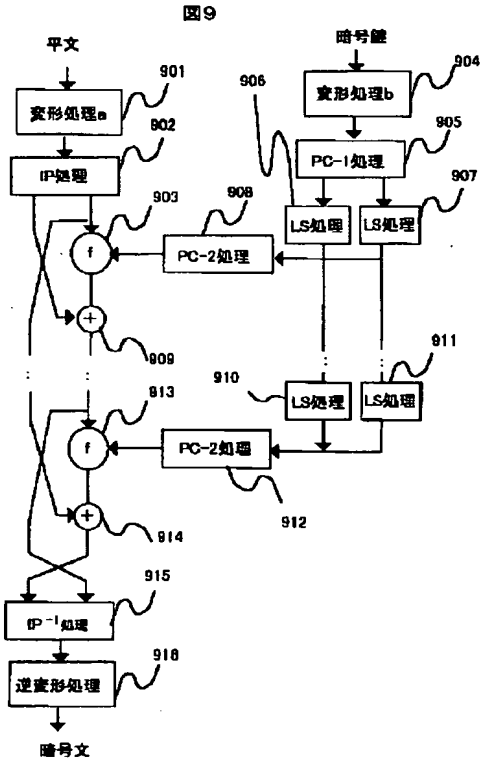

【図４】

図4



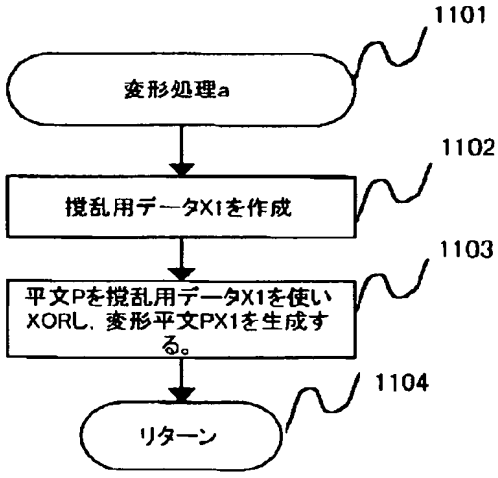

【図９】

図9



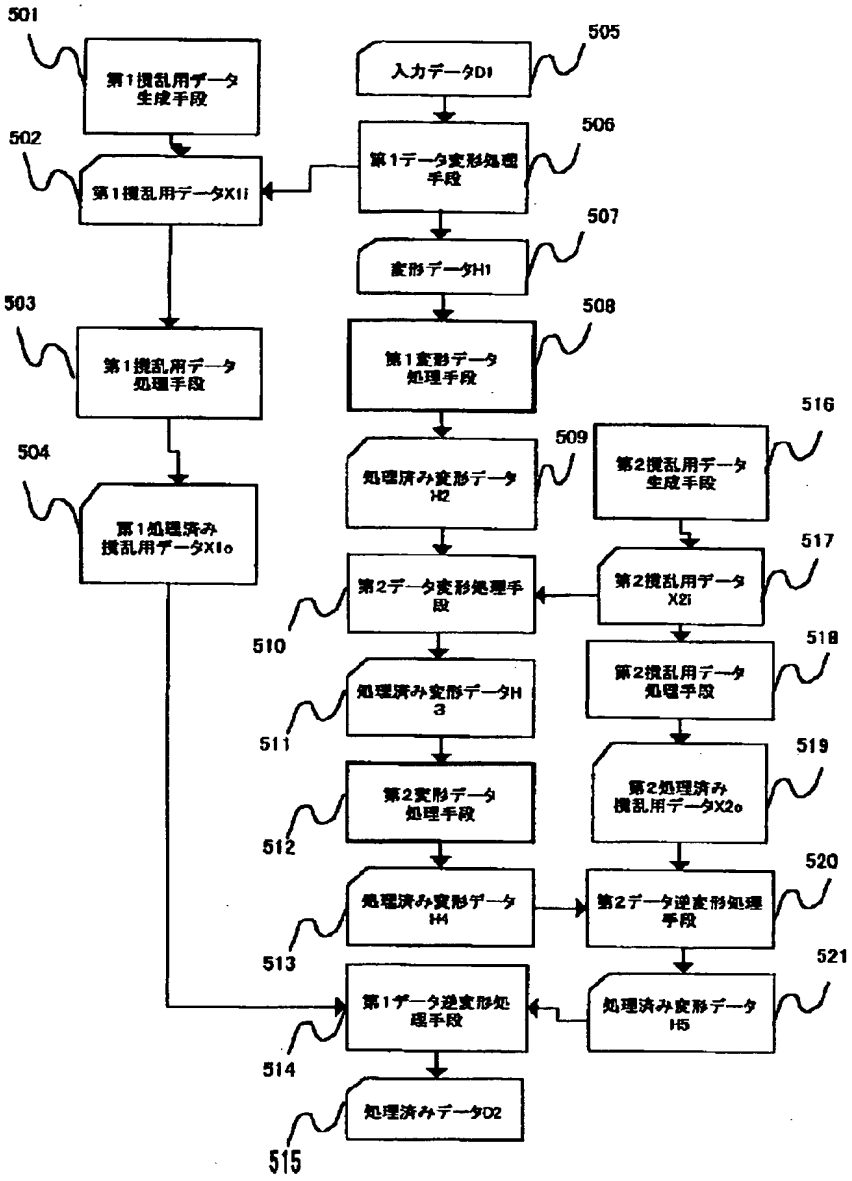

【図１１】

図11



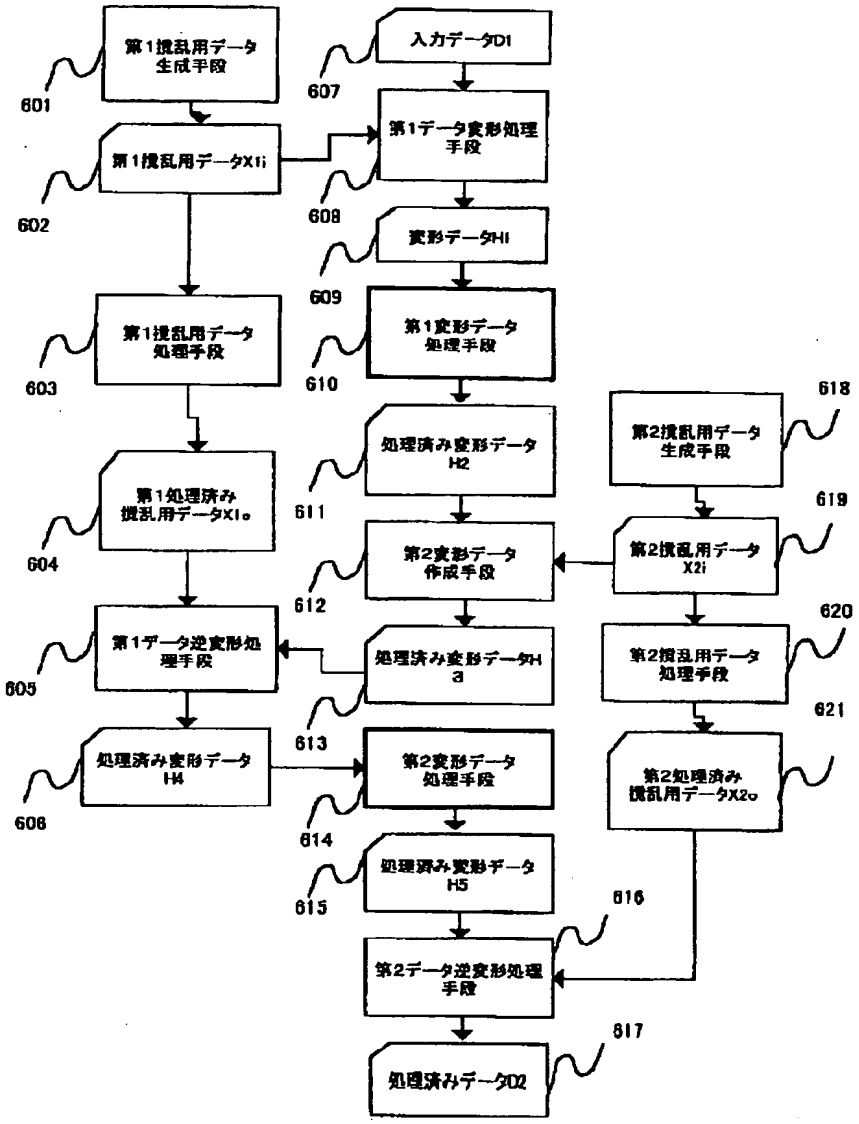
【図５】

図5


501
第1撹乱用データ生成手段
502
第1撹乱用データX1i
503
第1撹乱用データ処理手段
504
第1処理済み撹乱用データX1o
505
入力データDi
506
第1データ変形処理手段
507
変形データH1
508
第1変形データ処理手段
509
処理済み変形データH2
510
第2データ変形処理手段
511
処理済み変形データH3
512
第2変形データ処理手段
513
処理済み変形データH4
514
第1データ逆変形処理手段
515
処理済みデータDo2
516
第2撹乱用データ生成手段
517
第2撹乱用データX2i
518
第2撹乱用データ処理手段
519
第2処理済み撹乱用データX2o
520
第2データ逆変形処理手段
521
処理済み変形データH5

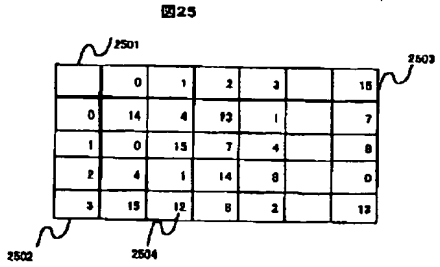
【図６】

図6


第1撹乱用データ生成手段
601
第1撹乱用データX1i
602
第1撹乱用データ処理手段
603
第1処理済み撹乱用データX1o
604
第1データ逆変形処理手段
605
処理済み変形データH4
606
入力データD1
607
第1データ変形処理手段
608
変形データH1
609
第1変形データ処理手段
610
処理済み変形データH2
611
第2変形データ作成手段
612
処理済み変形データH3
613
第2変形データ処理手段
614
処理済み変形データH5
615
第2データ逆変形処理手段
616
処理済みデータD2
617
第2撹乱用データ生成手段
618
第2撹乱用データX2i
619
第2撹乱用データ処理手段
620
第2処理済み撹乱用データX2o
621

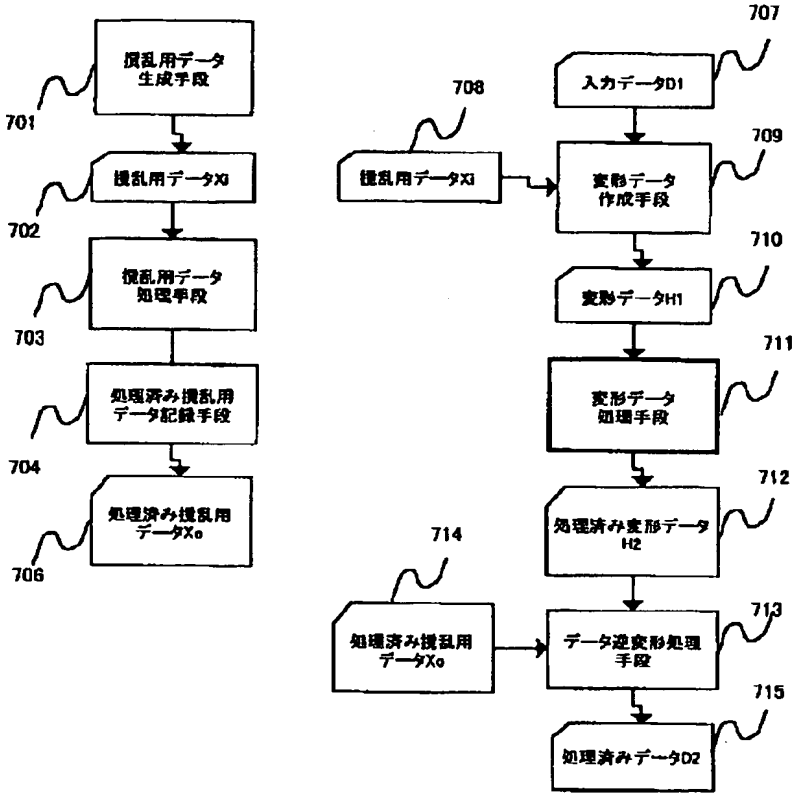
【図２５】

図25


2501
2502
2503
2504
0 1 2 3 15
0 14 4 13 1 7
1 0 15 7 4 8
2 4 1 14 8 0
3 15 12 8 2 13

【図７】

図7



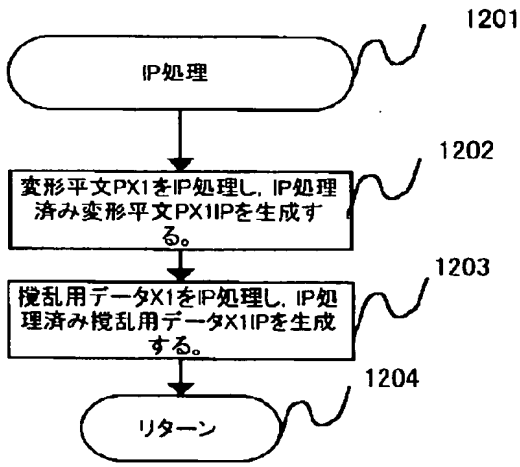

【図２９】

図29

| | | | | 2901 |
|---|---|---|---|---|
| 16 | 7 | 20 | 21 | |
| 29 | 12 | 28 | 17 | |
| 1 | 15 | 23 | 26 | |
| 5 | 18 | 31 | 10 | |
| 2 | 8 | 21 | 14 | |
| 32 | 27 | 3 | 9 | |
| 19 | 13 | 30 | 6 | |
| 22 | 11 | 4 | 25 | |

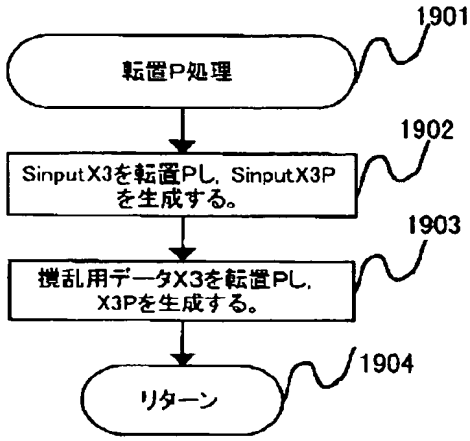
【図１２】

図12



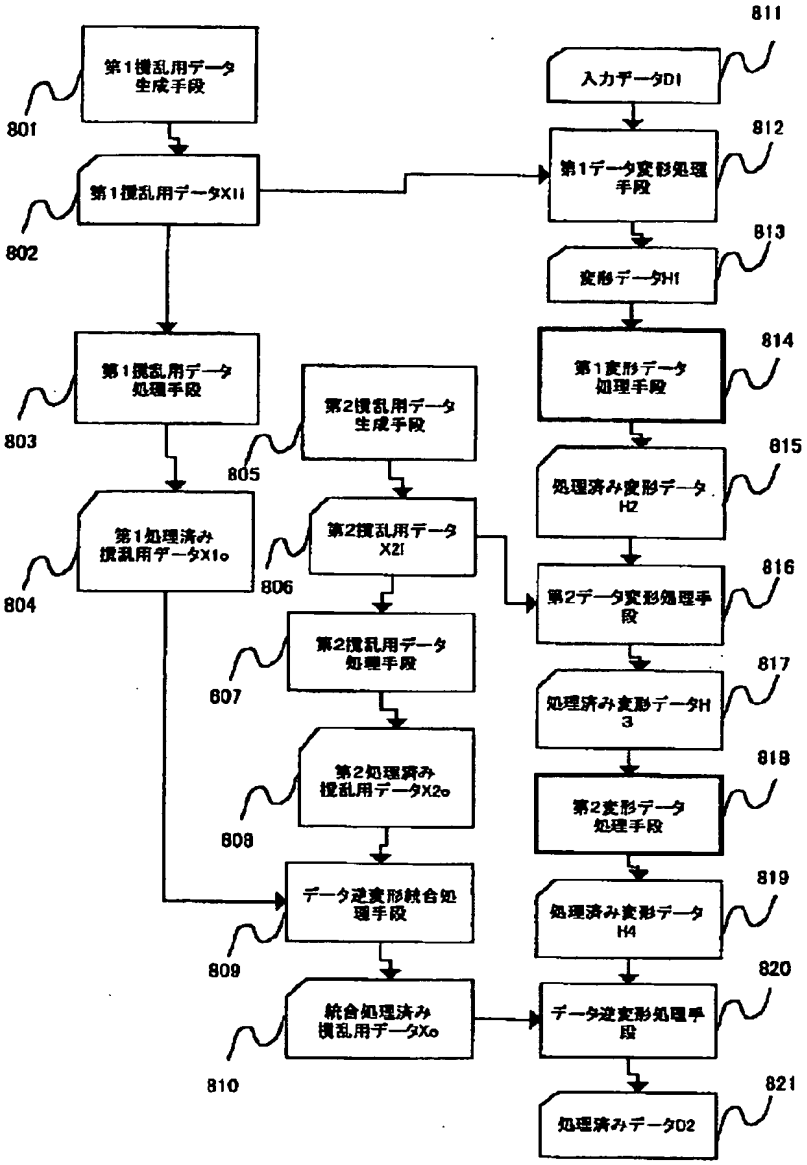

【図１９】

図19



【図８】

図8




【図３３】

図33

3301

| | 0 | 1 | 2 | 3 |
|---|---|---|---|---|
| 0 | B | 3 | 8 | 7 |
| 1 | 1 | 4 | 0 | D |
| 2 | 2 | F | A | 9 |
| 3 | 6 | C | 5 | E |

【図３４】

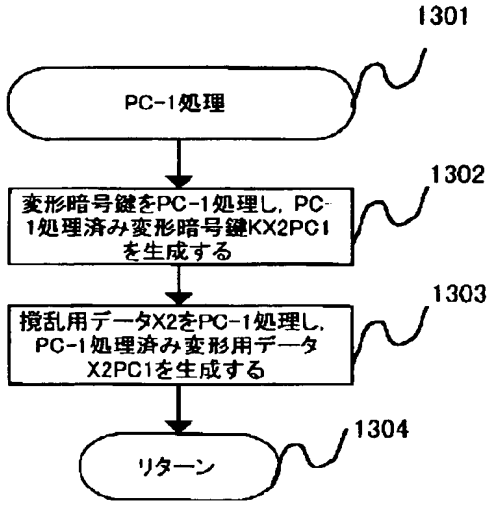
図34

3401

| | 0 | 1 | 2 | 3 |
|---|---|---|---|---|
| 0 | 2 | A | 1 | E |
| 1 | 8 | D | 9 | 4 |
| 2 | B | 6 | 3 | 0 |
| 3 | F | 5 | C | 7 |

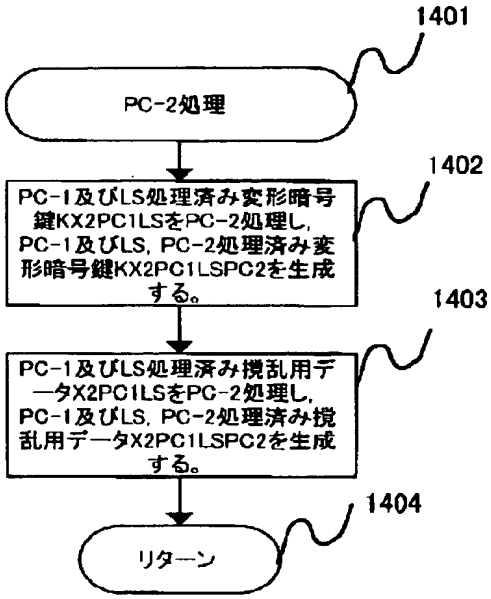
【図１３】

図13



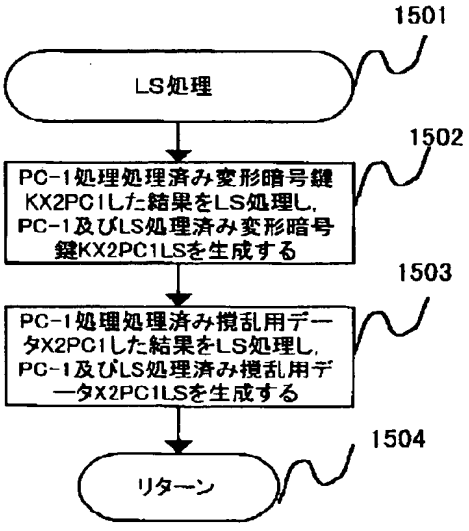

【図１４】

図14



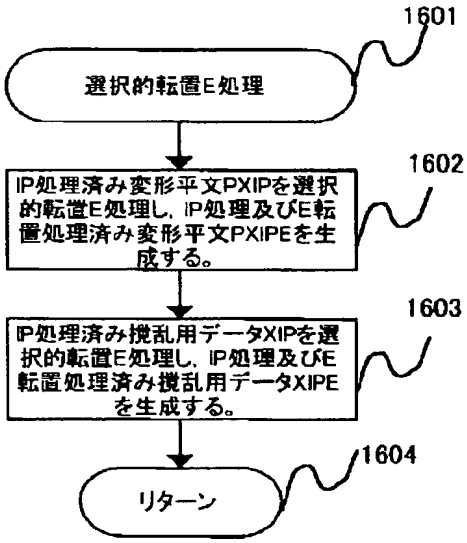

【図１５】

図15



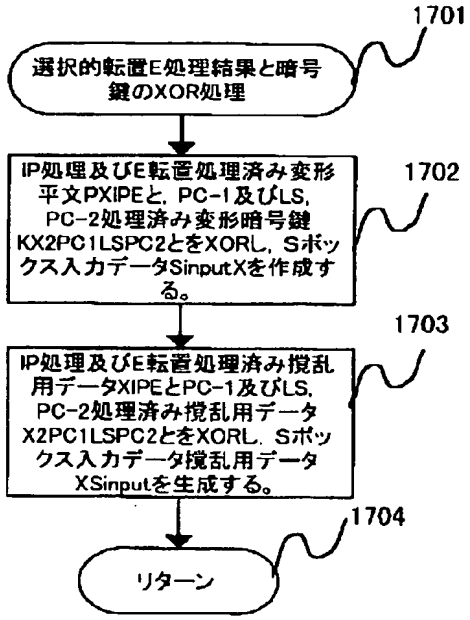

【図１６】

図16



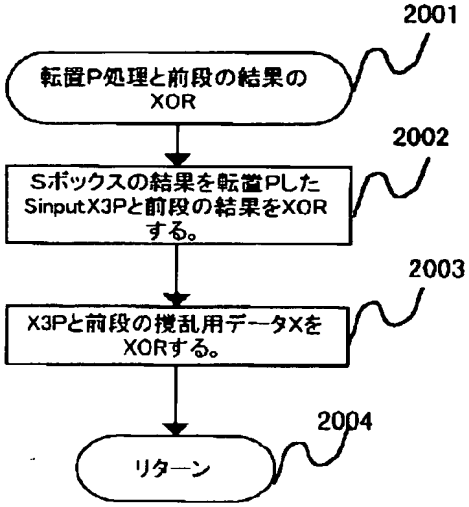
【図１７】

図17



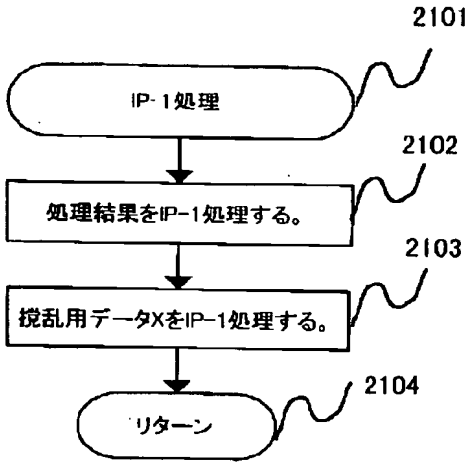

【図２０】

図20



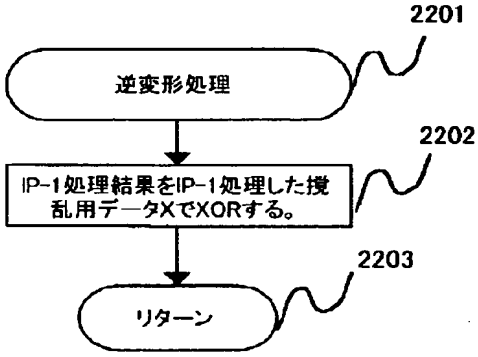

【図２１】

図21

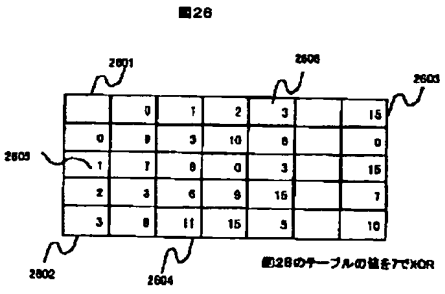



【図２２】

図22

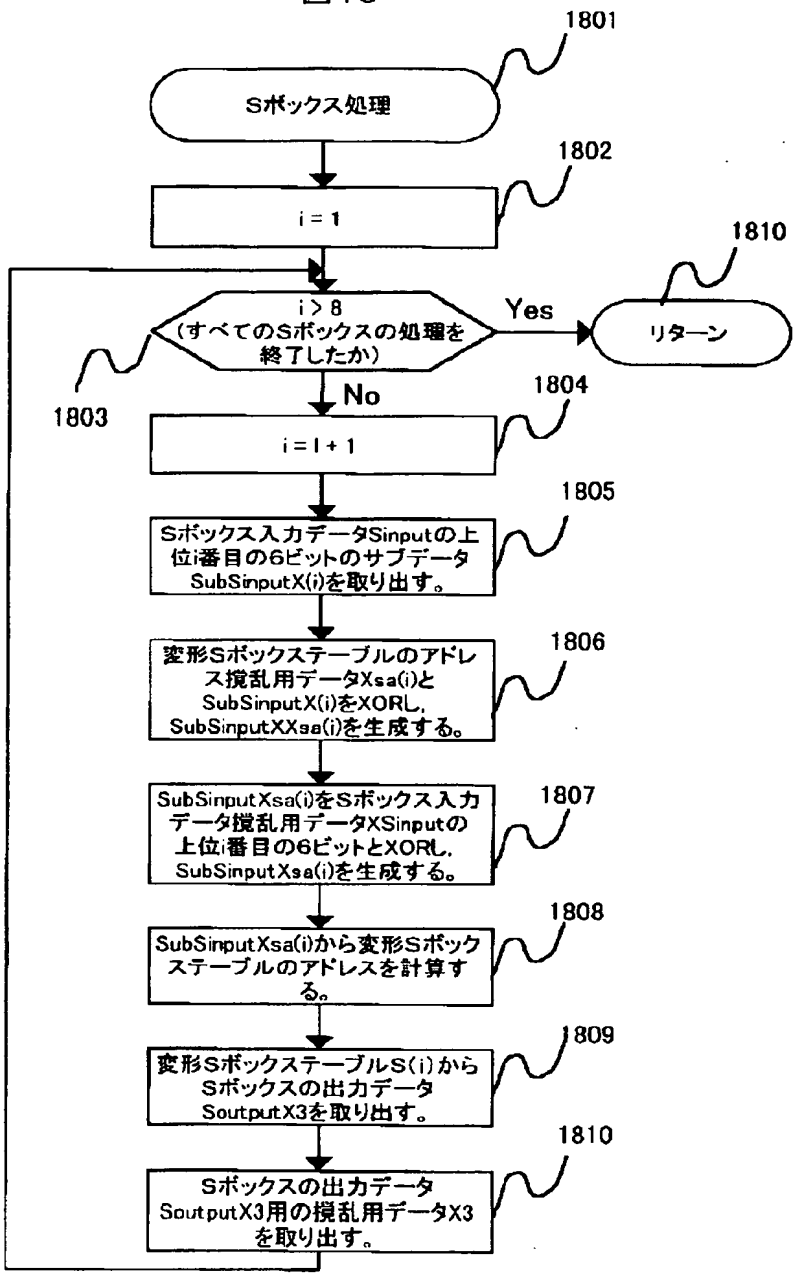



【図２６】



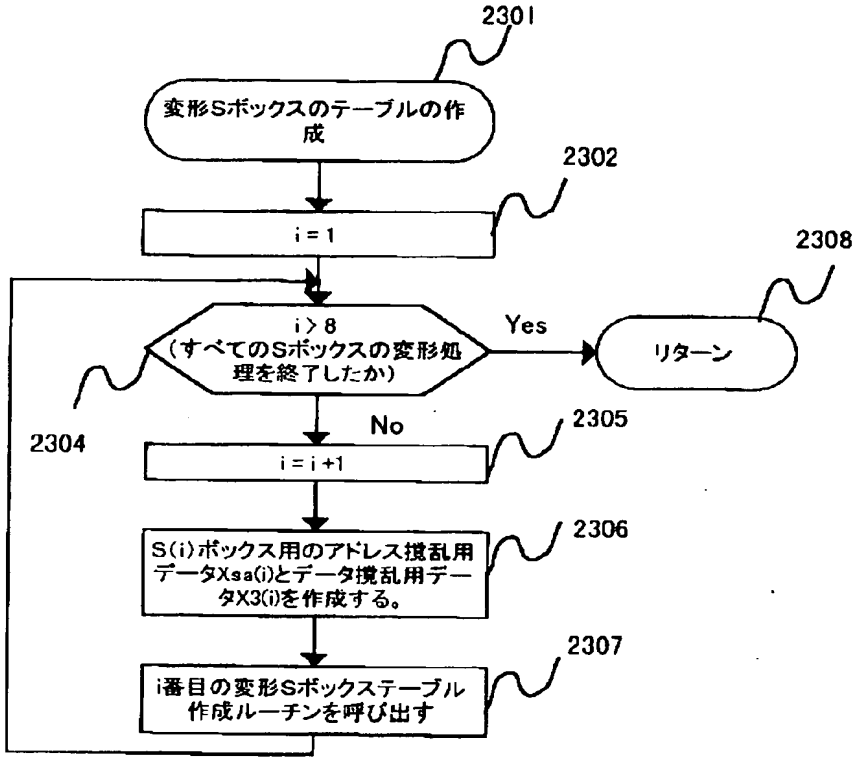
【図１８】

図18


1801
Sボックス処理
1802
i = 1
1810
i > 8
(すべてのSボックスの処理を終了したか)
Yes
リターン
1803
No
1804
i = I + 1
1805
Sボックス入力データSinputの上位i番目の6ビットのサブデータSubSinputX(i)を取り出す。
1806
変形Sボックステーブルのアドレス撹乱用データXsa(i)とSubSinputX(i)をXORし、SubSinputXXsa(i)を生成する。
1807
SubSinputXsa(i)をSボックス入力データ撹乱用データXSinputの上位i番目の6ビットとXORし、SubSinputXsa(i)を生成する。
1808
SubSinputXsa(i)から変形Sボックステーブルのアドレスを計算する。
1809
変形SボックステーブルS(i)からSボックスの出力データSoutputX3を取り出す。
1810
Sボックスの出力データSoutputX3用の撹乱用データX3を取り出す。

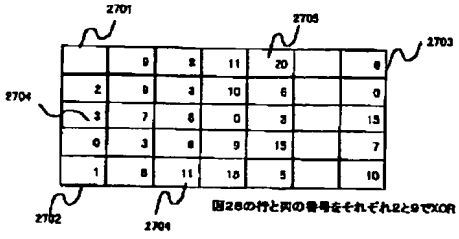
【図２３】

図23




【図２７】

図27



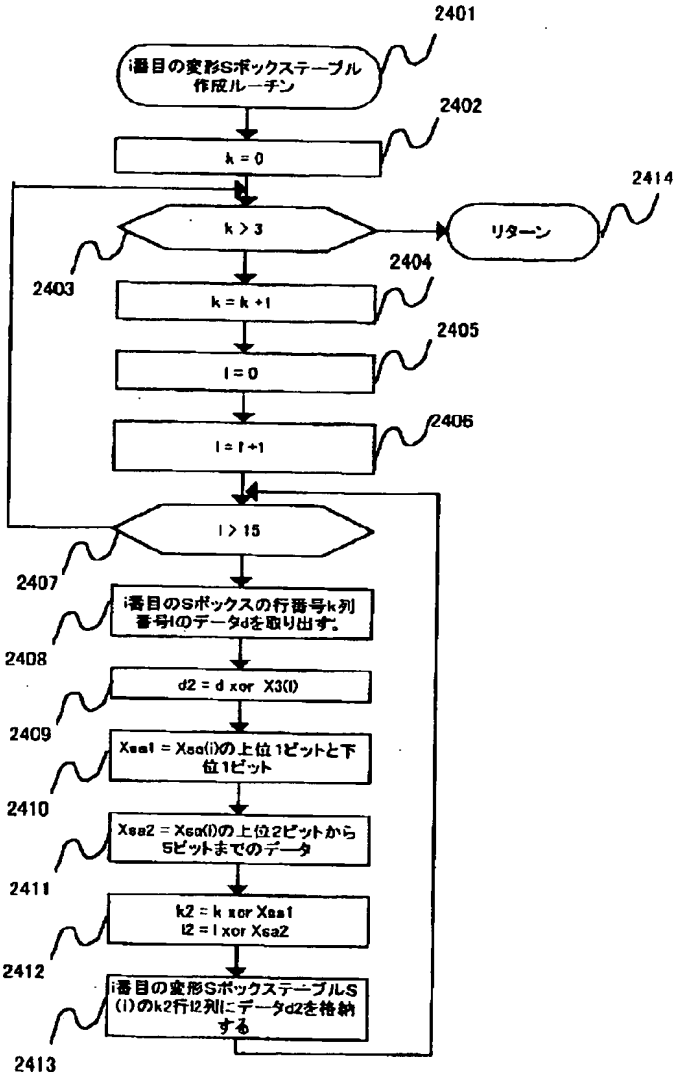

【図２８】

図28

| | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| 1 | 32 | 1 | 2 | 3 | 4 | 5 |
| 2 | 4 | 5 | 6 | 7 | 8 | 9 |
| 3 | 8 | 9 | 10 | 11 | 12 | 13 |
| 4 | 12 | 13 | 14 | 15 | 16 | 17 |
| 5 | 16 | 17 | 18 | 19 | 20 | 21 |
| 6 | 20 | 21 | 22 | 23 | 24 | 25 |
| 7 | 24 | 25 | 26 | 27 | 28 | 29 |
| 8 | 28 | 29 | 30 | 31 | 32 | 1 |

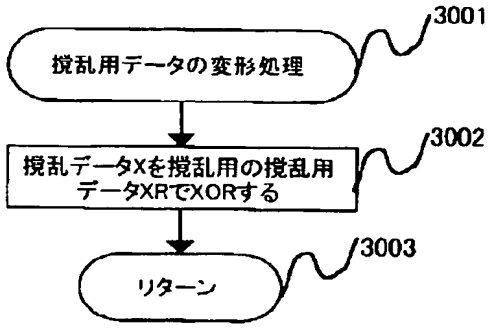
【図２４】

図24



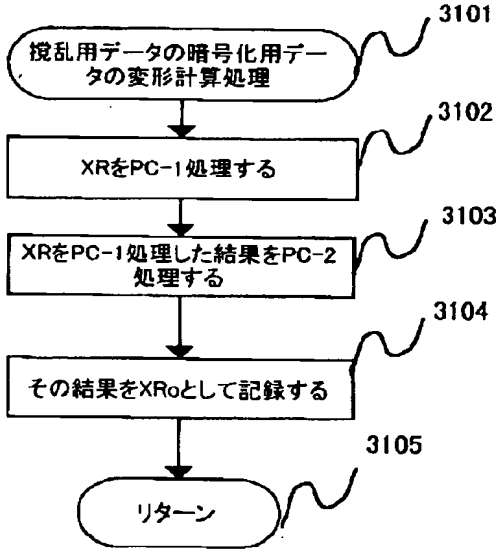

【図３０】

図30




【図３１】

図31




【図３５】

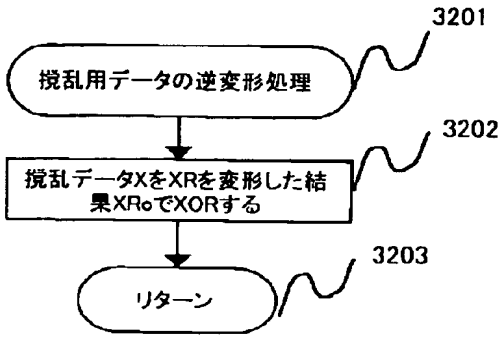
図35

| | 0 | 1 | 2 | 3 |
|---|---|---|---|---|
| 0 | 5 | F | 7 | C |
| 1 | 6 | B | 0 | 3 |
| 2 | D | 8 | 4 | 9 |
| 3 | A | 2 | E | 1 |

3501

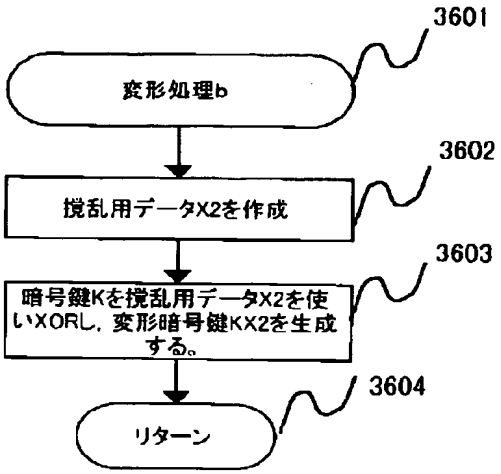
【図３２】

図32




【図３６】

図36




【図３７】

図37

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 58 | 50 | 42 | 34 | 26 | 18 | 10 | 2 |
| 60 | 52 | 44 | 36 | 28 | 20 | 12 | 4 |
| 62 | 54 | 46 | 38 | 30 | 22 | 14 | 6 |
| 64 | 56 | 48 | 40 | 32 | 24 | 16 | 8 |
| 57 | 49 | 41 | 33 | 25 | 17 | 9 | 1 |
| 59 | 51 | 43 | 35 | 27 | 19 | 11 | 3 |
| 61 | 53 | 45 | 37 | 29 | 21 | 13 | 5 |
| 63 | 55 | 47 | 39 | 31 | 23 | 15 | 7 |

【図３８】

図38

| | | | | | | |
|---|---|---|---|---|---|---|
| 57 | 49 | 41 | 33 | 25 | 17 | 9 |
| 1 | 58 | 50 | 42 | 34 | 26 | 18 |
| 10 | 2 | 59 | 51 | 43 | 35 | 27 |
| 19 | 11 | 3 | 60 | 52 | 44 | 36 |
| 63 | 55 | 47 | 39 | 31 | 23 | 15 |
| 7 | 62 | 54 | 46 | 38 | 30 | 22 |
| 14 | 6 | 61 | 53 | 45 | 37 | 29 |
| 21 | 13 | 5 | 28 | 20 | 12 | 4 |

フロントページの続き

(72)発明者　奥原　進
神奈川県横浜市戸塚区戸塚町5030番地　株式会社日立製作所ソフトウェア事業部内

(72)発明者　神永　正博
東京都国分寺市東恋ケ窪一丁目280番地
株式会社日立製作所中央研究所内

Ｆターム(参考)　5B035 AA13 BB09 CA12 CA38
5J104 AA01 AA47 JA03 NA35
9A001 BB02 EE02 EE03 FF04 GG01
LL03